Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2017

Rui Eduardo Lopes Carvalho

Sensor de Temperatura e Iluminação para Edifícios Inteligentes

Camera Sensor for *Smart Buildings*

Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2017

Rui Eduardo Lopes Carvalho

Camera Sensor for Smart Buildings

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Doutor Luís Filipe Mesquita Nero Moreira Alves, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro e do Mestre Nuno Lourenço.

o júri

presidente	Professor Doutor Pedro Nicolau Faria da Fonseca, Professor Auxiliar, Universidade de Aveiro
vogal	Professora Doutora Mónica Jorge Carvalho de Figueiredo, Professora Adjunta, Dep. de Engª Electrotécnica da Esc. Sup. de Tecnologia e Gestão do Inst. Politécnico de Leiria
vogal	Professor Doutor Luis Filipe Mesquita Nero Moreira Alves, Professor Auxiliar, Universidade de Aveiro

agradecimentos

Quero dedicar um especial agradecimento à minha família, principalmente os meus pais e irmão que sempre me suportaram durante esta fase tão importante da minha vida.

Aos meus amigos que com a sua amizade permitiram uma experiência memorável numa cidade que até então me era desconhecida.

À universidade de Aveiro que me deu uma oportunidade de obter uma educação incrível.

Um obrigado ao meu orientador Luís Alves pelo apoio dado na realização deste documento.

palavras-chave

Tecnologia CCD, Tecnologia CMOS, Sensores de Câmaras, Sistemas embutidos, Iluminação, Temperatura.

resumo

Com o aumento da prevalência das tecnologias IoT e a perseguição constante da automação de todas as ações simples do nosso dia-a-dia, a existência sensores simples e com pouco consumo de energia no nosso mundo tem tendência a tornar-se omnipresente. Uma área de estudo interessante seria como controlar eficientemente a luminosidade e a temperatura de uma sala.

A solução mais óbvia seria dispersar vários sensores equidistantes pela área a examinar, contudo a rede seria demasiado complexa do que seria expectável para um problema desta dimensão.

Neste caso quantidade não é uma resposta adequada para um problema com esta dinâmica. Um sensor único capaz de analisar uma área extensa é a solução ideal. Felizmente, com uma câmara é possível monitorizar a luminosidade e se tiver capacidade de deteção de infravermelhos também é possível controlar a temperatura.

Este documento examina a possibilidade e viabilidade de produção de um sensor único baseado numa câmara com tecnologia CCD ou CMOS com o objetivo de automatizar e regular a luminosidade e a temperatura de uma sala.

keywords

CCD Technology, CMOS Technology, Camera Sensors, Embbeded Systems, Lighting, Temperature.

abstract

With the increase in the prevalence of IoT technologies and constant pursue of automation of all simple actions in our day-to-day life, the existence of simple sensors and with little energy consumption have a tendency of becoming omnipresent. An interesting field of study would be how to efficiently control the lighting and the temperature of a room.

The most obvious solution would be to disperse several equidistant sensors through the area, however the network demand for such a task is too complex for a problem of this dimension.

In this case, quantity is not an adequate answer for a problem with this dynamic. A unique sensor capable of analyzing an extended area is the ideal solution. Fortunately, with a camera it is possible to monitor lighting, and should it have infrared detection capabilities is also possible to control the temperature.

This document examines the possibility and production viability of a unique sensor based on a camera with either CCD or CMOS technology with the objective of automate and regulate the luminosity and temperature of a room.

Index

Figure Index

Table Index

1. Introduction

Since time immemorial, mankind has always tried its best to put technology at service of progress. Whether we talk about revolutionary breakthroughs such as the wheel, the steam or simpler inventions like candleholders or electric toothbrushes, they are usually meant to make people's life easier (smooth life's wrinkles). When someone enters their home and with a few button presses they can relax in their sofa and enjoy a good TV show, it is all because technology was there to make it happen. As time passes, comfort demands are more exigent and new solutions are present every day to comply with this problem. One side effect of this reality is the increase of devices spread around us. However, since most of new gadgets nowadays have a need for a power supply, the power requirements of all environments grew and while raising the energy available is a viable option, it is not cheap. To try to keep this rise of power requirements to a minimum, energy saving became an important aspect when designing new technologies.

As the concern for both comfort and energy saving grows, the household market had to find new ideas. Smart environments tries to solve these issues.

1.1 Smart Environments

Smart environments take its roots in ubiquitous computing, which is a concept that heralds that computational devices may appear anywhere, no matter time and place. Its creator Mark Weiser stated that this was *"…a new field of computer science, one that speculated on a physical world richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives and connected through a continuous network."* [1]

By using IoT technology, homes are now capable of self-managing and by extension, improving the life of those living in them whether by controlling lighting, temperature or quality of air. New sensors are being developed every day to ensure an ever-growing positive experience while not forgetting about the need of low power consumption. As the number of computational devices

grows, in a world where everything is becoming interconnected, this field of ubiquitous computing is more of a reality in our everyday society.

However, not always quantity equals quality and an excessive number of devices spread within a smart environment will raise the network bandwidth to an inefficient cost-benefit ratio. For example, to fully map a room in terms of lighting and temperature, it would be needed several sensors from both requirements uniformly spread through the room. Still, this approach needs more communication resources than what would be expected for a project of this dimension. The best solution would include a single camera sensor that could cover both visible and infrared spectrums and get both lighting and temperature distributions through image analysis.

1.2 Objectives

The goal of this work is to design a sensor to monitor lighting and temperature based on a single CCD or CMOS camera. As referred above, the alternative approach would imply placing multiple devices distributed across a room which would require a bigger network than what's to be expected here.

In an automated environment, it is troublesome to constantly verify the lighting level in various sectors and adjusting the illumination system accordingly. In a similar manner, monitoring the temperature in order to make it uniform across a room is nearly impossible without an adequate number of sensors. Silicon based cameras can capture both the visible and the infrared spectrum and if they have a field of view large enough, can acquire information across a whole room. Through analysis of the visible spectrum, it is possible to obtain information about the lighting of an area, while the infrared part could be the key to unlock some data regarding temperature of that same area.

Two main camera technologies coexist which are the CCD and the CMOS. This document will discuss their differences and come with a conclusion of which side is the most suited to the aforementioned application.

After the device is chosen, continuous testing will be done to know if a camera with infrared capabilities is capable of solving this problem.

1.3 Structure of this document

This document is divided in 6 chapters. In the second chapter, some theoretical background is provided on radiometry and photometry, the electromagnetic spectrum, camera technologies, colorimetry and color spaces.

The third chapter delves into the architecture options for this work, the methodology used in the tests and the connection between the theory showcased in chapter two and its relevant applications in this work.

Chapter four tests the sensors capabilities and characteristics and checks its potential and limits.

Chapter five tests the sensors in a real scenario, how would they fare monitoring an area during a full day time.

Chapter 6 shows a summary of found conclusions and some ideas for future work.

2. Illumination and Cameras

2.1 Radiometry and Photometry

Electromagnetic radiation is made from waves that carry energy and the field created to study and define it is radiometry. Light is the part of the spectrum electromagnetic radiation as perceived by humans. Even though radiation can take form in all the wavelength spectrum, the human eye can only detect it in a limited band and has a nonlinear response. To further explore this matter, exists photometry [2].

2.1.1 Radiometry

Radiometry is the field that deals with the techniques required in the measurements of energy existent in any electromagnetic radiation. To explain it, some definitions are required:

Radiant energy (Q_e) is the quantity of energy that is transported by waves or particles, derived from electromagnetic radiation. This quantity is measured in joules (J) and considers all the wavelengths received (must be stated if otherwise).

Radiant flux (Φ_e) or radiant power describes the flow of energy per unit time:

$$\phi_e = \frac{dQ_e}{dt} \tag{1}$$

The units of this parameter are joules per second, which is equivalent to watts (1 J/s = 1 W).

It is possible to measure the radiation emitted by a body using a device that produces an electric current proportional to the magnitude of said radiation. It is correlated with the radiant flux.

Irradiance (E_e) or radiant flux density, is the radiant flux per unit area in each surface. All directions are considered. Mathematically, it is defined by:

$$E_e = \frac{d\phi_e}{dA} \tag{2}$$

Where Φ_e is the radiant flux and A is the area of surface analyzed and its units are watts per square meter (w/m^2). When radiation leaves a surface, it is called exitance (M_e).

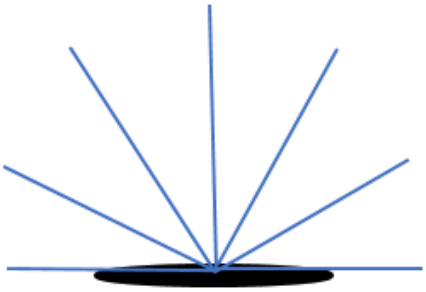

Figure 1: Irradiance/exitance is the sum of radiation (the blue lines) leaving/exiting a surface of a certain area (the black body)

Radiant intensity (I_e) is defined as the radiant flux emitted (or incident) in a certain solid angle of a body. Therefore, its units are W/sr and is calculated using radiant flux (Φ_e) and the solid angle (ω) of the direction of analysis:

$$I_e = \frac{d\phi_e}{d\omega} \tag{3}$$

Radiance (L_e) is the amount of radiant flux in an infinitesimal ray arriving or leaving a determined point in a determined direction of a body. Its units are W/m^2/sr and is defined by this equation:

$$L_e = \frac{d^2\phi_e}{d\omega\, dA cos\theta} \tag{4}$$

φ is the radiant flux, dA is the area containing the point in which the ray hits, dω is the solid angle of the direction the radiation is travelling and θ is the angle between this direction and the surface normal angle. $dAcos\theta$ is called the projected area.

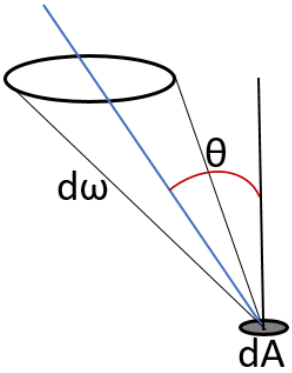

Figure 2: Radiance, a ray of radiation (blue line) leaving a surface of size dA meters at an angle of θ with a spread of dω steradian.

All these quantities may be analyzed at a certain wavelength interval λ and can be described mathematically with these expressions:

Quantity	Expression	Units
Spectral Radiant Energy	$Q_{e\lambda} = \frac{dQ_e}{d\lambda}$	J/m
Spectral Radiant Flux	$\phi_{e\lambda} = \frac{dQ_{e\lambda}}{dt} = \frac{d\phi_e}{d\lambda}$	W/m
Spectral Irradiance	$E_{e\lambda} = \frac{dE_e}{d\lambda} = \frac{d^2Q_{e\lambda}}{dA\,dt} = \frac{d^2\phi_e}{dA\,d\lambda}$	W/m^3
Spectral Radiant Intensity	$I_{e\lambda} = \frac{dI_e}{d\lambda} = \frac{d^2Q_{e\lambda}}{d\omega\,dt} = \frac{d^2\phi_e}{d\omega\,d\lambda}$	W/sr/m
Spectral Radiance	$L_{e\lambda} = \frac{d^3\phi_e}{d\omega\,dA cos\theta\,d\lambda}$	W/sr/m^3

Table 1: Spectral equations for radiometry

Note that when wavelengths are limited, radiometric unit become spectral radiometric units, radiant energy Q_e becomes spectral radiant energy $Q_{e\lambda}$, irradiance E_e is now spectral irradiance $E_{e\lambda}$, etc.

2.1.2 Photometry

Radiometry provides a broad understanding of the energy contained in electromagnetic radiation, but for some applications it is not enough. Humans eye photodetectors have nonlinear sensitivity and thus, detect radiation wavelength in a peculiar way. Photometry exists to fill this gap as it studies the measurement of light as perceived brightness for people.

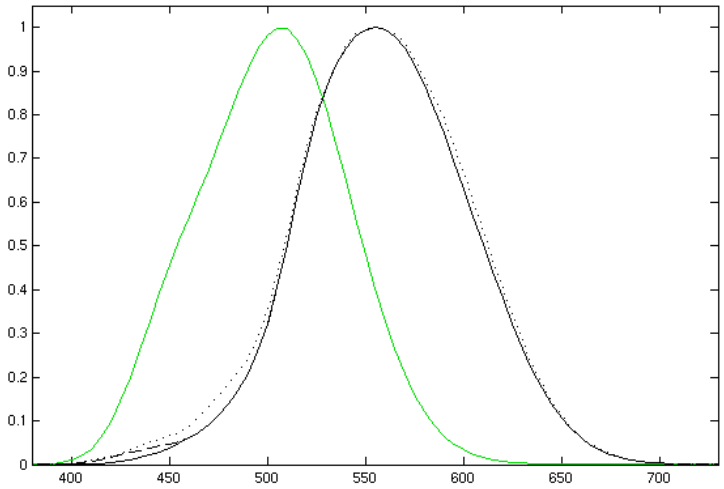

Figure 3: Spectral efficiency for the human eye, (green line is called scotopic and black line is photopic) [4]

For brightly lit places, humans detect wavelengths ranging from 380 to 770 nanometers, with a peak perception for 555 nm. In this scenario, the photopic luminosity function is active but, as radiant energy becomes scarce, the eye will automatically adapt accordingly and this will no longer be valid. Instead the sensitivity will shift accordingly to lower wavelengths and at dimly lit places, the response would be like the scotopic curve.

The most important SI unit in photometry is the candela (cd), defined as the luminous intensity, in a given direction, of a source that emits monochromatic radiation of frequency 540 THz (555 nm) and that has a radiant intensity in that direction of 1/683 watt per steradian [3]. Important units derived from the candela are the lumen (lm), being candela times steradian (cd·sr) and the lux (lx) which is lumen per square meter (lm/m^2).

To obtain the photometric quantities, it is needed to weight the radiometric correspondent by the adequate luminosity function (photopic or scotopic).

$$\Phi_v = \int \Phi_e(\lambda) * V(\lambda)\, d\lambda \tag{5}$$

Where Φ_e is the radiant flux, V is the photopic curve and Φ_v is the luminous flux.

Photometric quantity	**Radiometric correspondent**	**Units**
Luminous Intensity (I_v)	Radiant Intensity	cd
Luminous Flux (ϕ_v)	Radiant Flux	lm (cd.sr)
Luminous Energy (Q_v)	Radiant Energy	lm/s
Illuminance (E_v)	Irradiance	lx (lm/m^2)
Luminance (L_v)	Radiance	cd/m^2

Table 2: Photometric quantities

Another important unit is the luminous efficacy (K) which measures how efficiently a light source produces visible light. Its mathematical formula can be given by the luminous flux (Φ_v) divided by the radiant flux (Φ_e):

$$K = \frac{\Phi_v}{\Phi_e} \tag{6}$$

Luminous efficacy can have different meanings depending on the context. While its unit is always lumens per watt (*lm/W*) and the lumens components always comes from the luminous flux provided, the watt may refer to either the radiant flux (in this case it is the ratio between total and visible electromagnetic radiation emitted as expressed on equation 6) or it can be the power consumed by the light source. The former is called luminous efficacy of radiation and the latter luminous efficacy of a source.

As lighting efficiency is becoming even more of a concern, with multiple types of light bulbs and LEDs in the market, the luminous efficacy started to be used as a unifying unit as its value will depend on the technologies used:

Light type	Typical luminous efficacy (lumens/watt)
Tungsten incandescent light bulb	12.5-17.5 lm/W
Halogen lamp	16-24 lm/W
Fluorescent lamp	45-75 lm/W
LED lamp	80-100 lm/W
Metal halide lamp	75-100 lm/W
High pressure sodium vapor lamp	85-150 lm/W
Low pressure sodium vapor lamp	100-200 lm/W
Mercury vapor lamp	35-65 lm/W

Table 3: Luminous efficacy in different light sources [5]

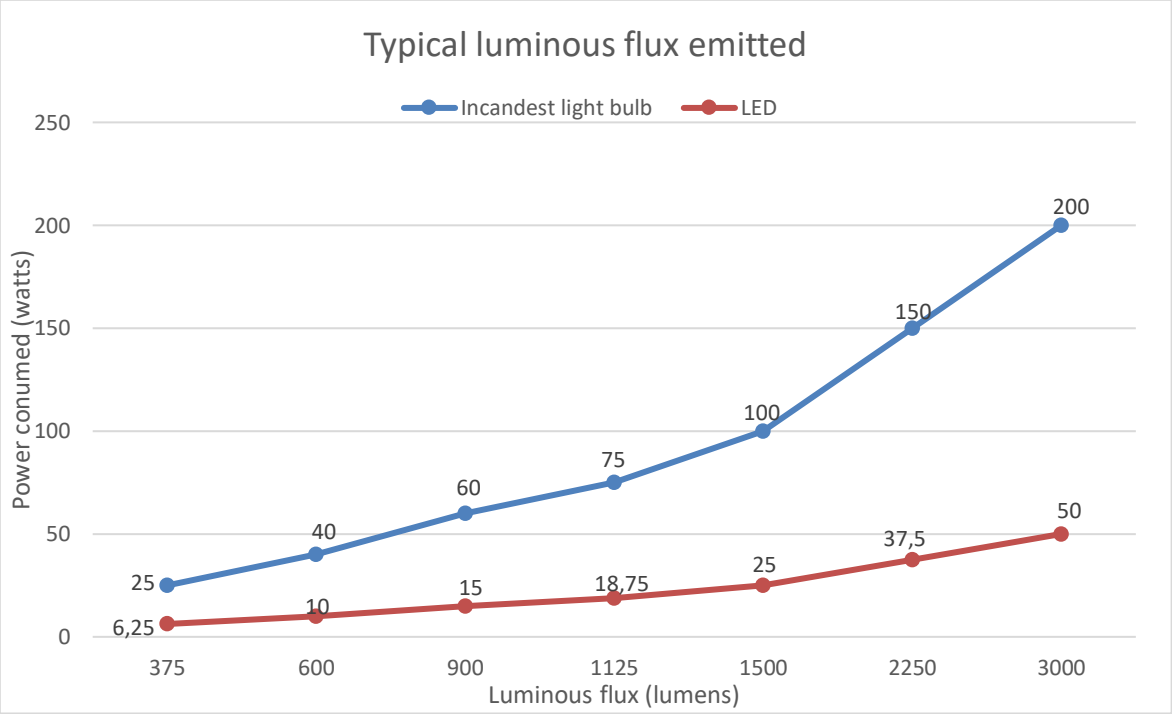

Figure 4: Typical lumens emitted by incandescent light bulbs and LEDs [6]

Should we take the above example featuring incandescent light bulbs and LEDs, with the same power usage, LEDs offer four times more lighting potential.

Photometric quantities are also relevant in the measurement of the light level in a room and by extension when planning the light sources distribution in our homes. Even though personal preference plays a big part in how we use and distribute light sources, guidelines exist, because not all premises require the same illumination and not everyone is a lumen expert to understand if 4 uniformly distributed LEDs worth 500 lumen each is enough for our 16-square meter bedroom.

Premise	Typical Illuminance requirements (lux)
Halls and conference rooms	500 lx
Classrooms	300 lx
Stairs in public buildings	75 lx
Bedroom	150 lx
Stadium	750 lx
Office	300 lx
Kitchen	100 lx

Table 4: Light level requirements in different premises [6]

Table 4 provides some sample data in how much light it is needed in certain premises. A bedroom would require 150 lux or 150 lumens per square meter. A 16-square meter bedroom needs 150*16=2400 lumens to have efficient illumination. The person in the above example should add a fifth luminaire just to be sure that he would get enough light in his room.

2.2 Temperature-Infrared Spectrum Relationship

The electromagnetic spectrum refers to all the frequencies known and their corresponding wavelengths of radiations. The radiation with the lowest wavelength know are the gamma rays and have just one picometer. A great point of interest are the wavelengths contained in the visible spectrum. Starting at 400 nanometers and stopping at 700 nanometers it provides all the colors humans can see. Any wavelengths beyond these limits will not be recognized by our eye receptors, therefore are not detected. To detect them, cameras are required in order to correctly map the

amount of energy contained in the wavelengths in our surroundings. This way, after adequately processed, information can be presented in a way to allow human comprehension.

The infrared spectrum is way larger than the visible one. It goes from 700 nanometers until 1 millimeter. Since infrared includes a lot of wavelengths it is usually divided further in sub-divisions as seen in table 5 [22]:

Sub-Division	Wavelength
Near-Infrared (NIR)	0.78-3μm
Mid-Infrared (MIR)	3-50μm
Far-Infrared (FIR)	50-1000μm

Table 5 – Infrared ISO 20473 division scheme

The exact intervals may differ slightly according to the scheme used, but for the sake of clarity, we will stick to the ISO 20473 division.

Even though they are all infrared radiation, each sub-division spectroscopy is useful in a different field. Applications for the NIR spectrum focus on medical research for non-invasive diagnoses of diseases or alternative ways to measure blood sugar because NIR is minimally absorbed by water and hemoglobin [24]. Other applications include analysis of plant health through the amount of chlorophyll present [25], atmospheric chemistry or combustion research. Most common cameras also detect NIR to a certain degree, making the analysis of both visible light and NIR with a single camera easier.

The Mid-Infrared is where thermal radiation is situated at room temperature and it can be shown through the Wien's displacement law [17]. This law states that the radiation curve for different temperatures peaks at a wavelength inversely proportional to the temperature. The formula to discover at which wavelength the spectral energy peaks to a given temperature is given by:

$$\lambda_{max} = \frac{b}{T} \tag{7}$$

With b being the Wien's displacement constant, equal to 2.8977729×10^{-3} m·K, λ_{max} is the wavelength where the peak occurs and T is the temperature measured in Kelvins.

In this work, it will be assumed that all objects will be black bodies, meaning that all electromagnetic radiation will be absorbed. The black body radiation spectrum is defined by Planck's law given by:

$$S(\lambda) = \frac{2hc^2}{\lambda^5}\left[e^{\frac{hc}{\lambda kT}} - 1\right]^{-1} \quad (8)$$

In this equation, h is Planck's constant, c is the speed of light, k is the Boltzmann's constant, T the temperature in Kelvin degrees and λ the wavelength. Figure 5 shows some examples of this formula.

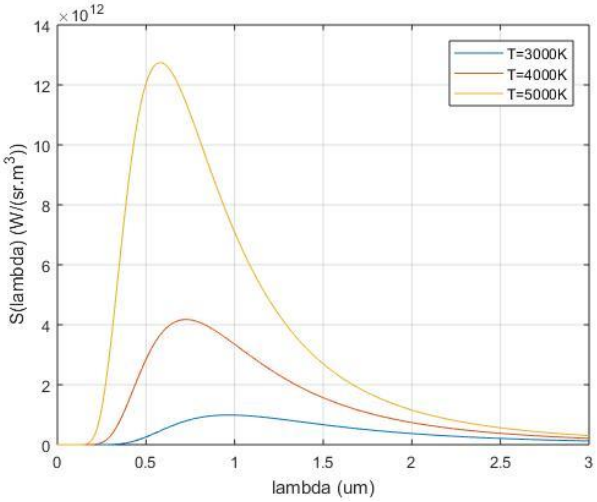

Figure 5: Black body radiation at 3000K, 4000K and 5000K.

According to 5, at 4000K or higher, the body would start emitting visible light. At room temperature the story is different. A black-body at room temperature would have around 300 Kelvins (27ºC). Through the Wien's displacement law, it can be shown that its peak wavelength is at 9.66μm.

And Planck's law gives its full temperature radiation emission:

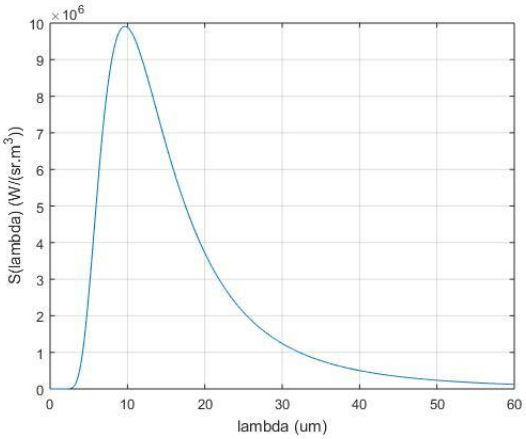

Figure 6: Black body radiation at 300K.

When analyzing a room, there won't be any perfect black-bodies, however it is safe to assume that a camera should have mid-infrared capabilities to be able to map the temperature in a room.

2.3 Photoelectric effect

The photoelectric effect is the emission of electrons when light hits a material. Even though they are normal electrons, they can be called photoelectrons, to differentiate their origin. It was observed by Heinrich Hertz in 1877 who saw that when certain frequencies of light were focused onto a metal, it would exhibit a spark. J.J. Thomson identified these sparks as electrons, who were excited by the beam of light and left the surface of the metal. [9]

The physicists of that time, using classical electromagnetic theory, to explain this phenomenon began working on the model that the light traveled through space as an indivisible wave and when it hit a metal, it would transfer part of its energy onto the electrons over time, making them vibrate. When they had enough energy, they would be released from the material, becoming free electrons. With the knowledge they had at the time they predicted that higher light amplitude should increase the kinetic energy of photoelectrons and the rate of emission (measured through electric current) should increase alongside the frequency of the wave. Unfortunately, nothing of the sort happened! The kinetic energy increased with light frequency, while the electric current increased with light amplitude, instead.

A new model was needed and Albert Einstein provided it. He proposed that light travels not as a wave, but as discreet particles called photons. Max Planck provided a handy equation that dictated the energy a photon was carrying. The formula is $E = hv$, with h being Planck's constant and equal to $6{,}626\times10^{-34}$ J·s and v is the frequency in Hz. This means that the energy (E in joules) is proportional to the frequency. [9]

Returning to the matter of the photoelectric effect, the scientists also discovered that if a beam had insufficient frequency, no electrons would be liberated, resulting in the received energy being re-emitted. The minimum energy in which the photoelectric effect occurs depends on the material targeted and is called the work function (Φ).

Material	Work function Φ (Joules, J)
Calcium (Ca)	4.60×10^{-19}
Tin (Sn)	7.08×10^{-19}
Sodium (Na)	3.78×10^{-19}
Hafnium (Hf)	6.25×10^{-19}
Samarium (Sm)	4.33×10^{-19}
Silicon (Si)	7.36×10^{-19}

Table 6: Example of work functions

When an electron is removed, the remaining energy goes to its kinetic energy:

$$KE_{electron} = E_{photon} - \Phi \quad (9)$$

2.4 Camera Technologies

2.4.1 Silicon

Silicon is not used in imagers industry by coincidence. It is widely used in a multitude of areas as it a prime material for the construction of integrated circuits and therefore has a wide and active market behind it. It brings a good advantage in acquiring this component in the making of imagers. The main reason however is silicon's optical properties. Usually cameras are used at ambient temperatures and must be sensible at the visible light spectrum (400nm to 750nm). Silicon has a high absorption rate of photons at these wavelengths:

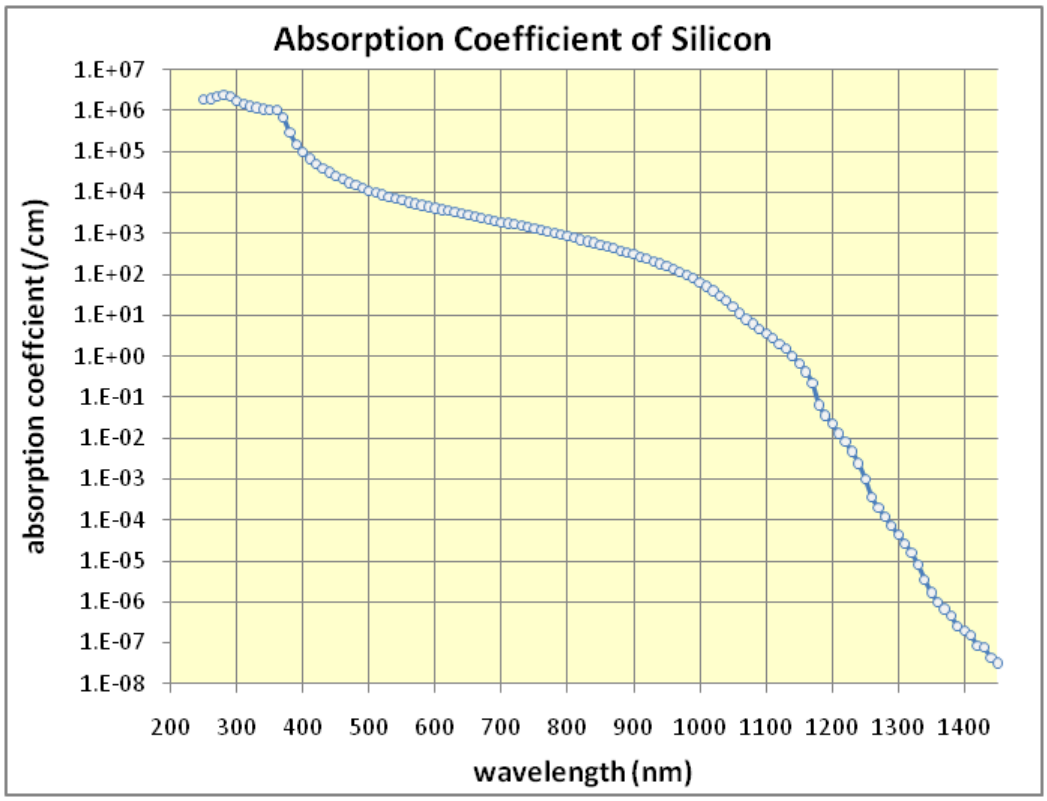

Figure 7- Absorption coefficient of silicon as a function of the wavelength [7]

Because of this, silicon will readily absorb the photons at the important wavelengths, which will translate into excited electrons and an electric stimulation required for the output of an image. Notice that the coefficient is significant in the near infrared spectrum (it sharply drops at 1100 nanometers though). This means that an imager based on silicon will have infrared capabilities up to a certain degree. To prevent these wavelengths from appearing in a common camera, an infrared filter is usually shipped together with the imager so that only the visible light spectrum is captured by the sensor.

2.4.2 CCD and CMOS Technology

Charge coupled devices (CCD) and complementary metal oxide semiconductor (CMOS) are two competing imager technologies [8]. They both use the principle of the photoelectric effect to operate and their sensors are divided by pixels. These pixels are, in the case of the CCD, formed by an epitaxial layer of silicon and transistors. Photons strike the silicon surface, forming free electrons through the photelectric effect. To counteract this, a transistor bellow is positively biased. Because there is an insulator between the gate and the electrons, they can't join and become trapped in place while the photoelectric effect occurs. After the exposure time ends (the time where

the silicon is bombarded with photons), the bias is transferred to a nearby gate, causing the trapped electrons to follow it. This process repeats, until it reaches the end of the image sensor, where they are converted to voltage and shipped off sensor to an auxiliary camera board [23].

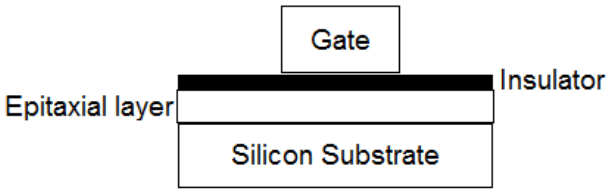

Figure 8 – Diagram of a CCD pixel

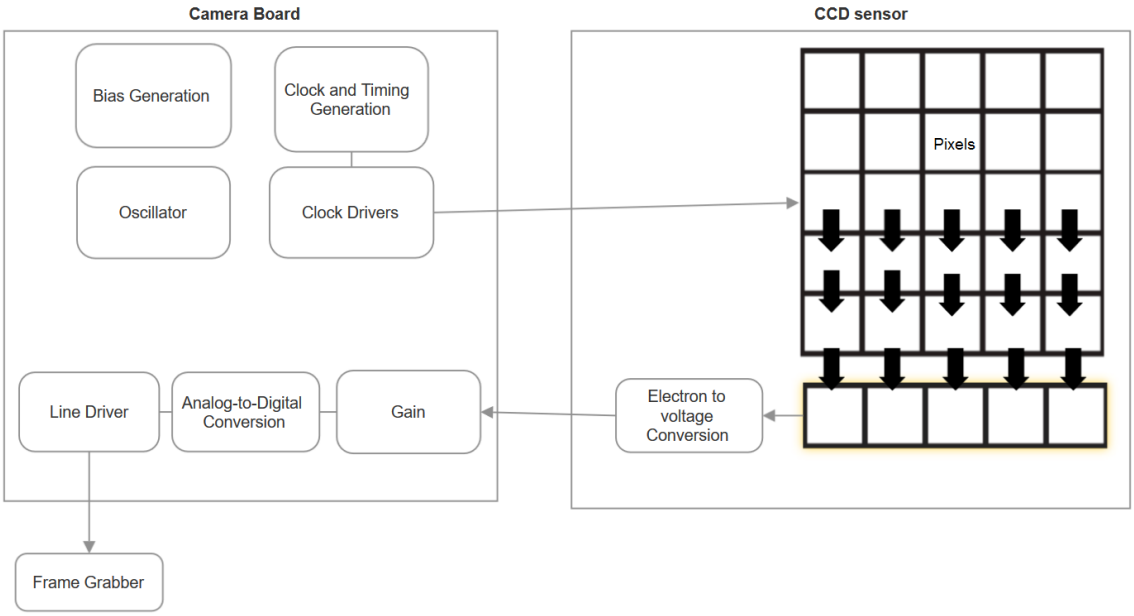

Figure 9 – Diagram of a CCD camera

After arriving at the camera board, they will have a gain floor, will be converted to digital in order to be compatible with external devices such as computers or monitors, and then a frame grabber collects the data.

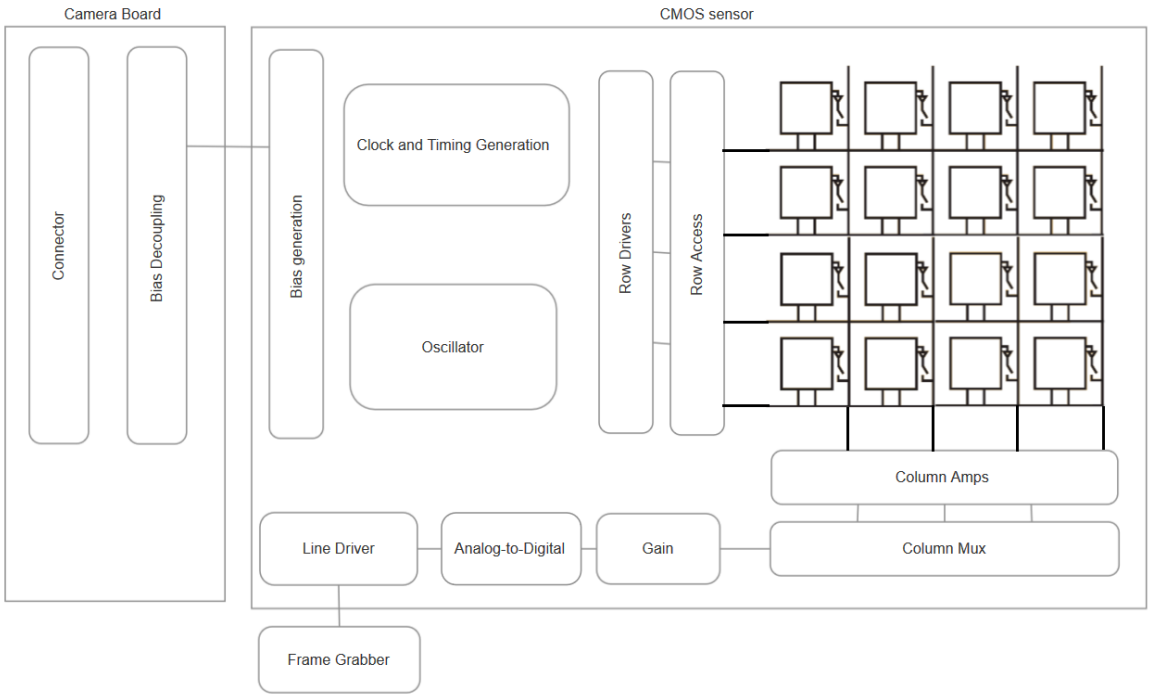

Figure 10 – Diagram of a CMOS camera

The CMOS functions differently. Instead of storing electrons and converting them at the edge of the sensor, they have integrated circuits right on top the pixels that allows immediate electron-voltage conversion. By having extra electronics in the pixels, the CMOS has less capture area for photons, thus dropping a bit their efficiency. While the CCD had the majority of its electronics outside the sensor, the CMOS is the opposite making it more difficult to change any auxiliary electronics. Some of their differences are resumed bellow:

- The CMOS have better responsivity than CCD technology, meaning that their sensor has better output signal per input optical energy.
- Regarding the dynamic range, the ratio of a pixel's saturation level to its signal threshold, the CMOS are worse because of all the noise their embedded circuitry in the imager produces and less access to external cooling.
- CMOS also perform a bit poorly in the ability to produce the same output with the same conditions (uniformity), as having an amplifier in each pixel with slight offset variants can cause poor uniformity.
- Because all circuitry is directly on the CMOS imager, they process the data faster.
- In general, CCDs have better image quality and have a greater size.
- The biggest advantage though is that CMOS has better market availability and cheaper prices.

In summary, the CCD technology brings more stability and superior image quality, while CMOS are faster, cheaper and smaller which are critical characteristics to the project. [8]

2.4.3 Sensor characteristics

When choosing a camera, with so many different options in the market, the buyer should know what he is looking for. Some key characteristics are the size, cost, field of view (FoV), sensitivity, cooling and software available.

Different sizes will affect the uses of a camera. Different mounts, lens and other peripherals will fit and a big camera will not be as adaptable as a small one. As in everything, cost is a defining factor when choosing the right camera, since budget is limited. The number and size of pixels affects the image recorded by a camera. Having a greater number of pixels will increase the field of view with little drawbacks (except the higher price). Increasing the size of each individual pixel will also augment the field of view but at the cost of a lower image resolution.

The sensitivity is usually defined by a lot of variables, but it can be easily summarized in the quantum efficiency of the sensor. Simply put, the quantum efficiency measures the efficacy of the device in turning incoming photons to electrons. This is important, as the conversion is different for each wavelength and the brightness is determined by the number of electrons in a pixel well. The wells have a limit of how many electrons they can store and if oversaturated, they can contaminate nearby pixels (called blooming).

The software available to manage the camera via a computer is also very important. A good software will allow the manual tuning of the camera to adapt according to the environment and process the image in the best way possible.

2.5 Colorimetry

Colorimetry is the science that describes and allows the quantification of the human color perception and links it to the visible spectrum. The most used model is the CIE 1931 color space, who introduced the X, Y, Z tri-stimulus system, based on the primary colors red, blue and green. The calculation method requires the spectral radiance distribution ($L_e(\lambda)$) to be weighted with the human eye response curves $\bar{x}(\lambda), \bar{y}(\lambda), \bar{z}(\lambda)$ [14] and integrated afterwards. Therefore, their equations are:

$$X = \int L_e(\lambda) * \bar{x}(\lambda)\, d\lambda \tag{10}$$

$$Y = \int L_e(\lambda) * \bar{y}(\lambda)\, d\lambda \tag{11}$$

$$Z = \int L_e(\lambda) * \bar{z}(\lambda)\, d\lambda \tag{12}$$

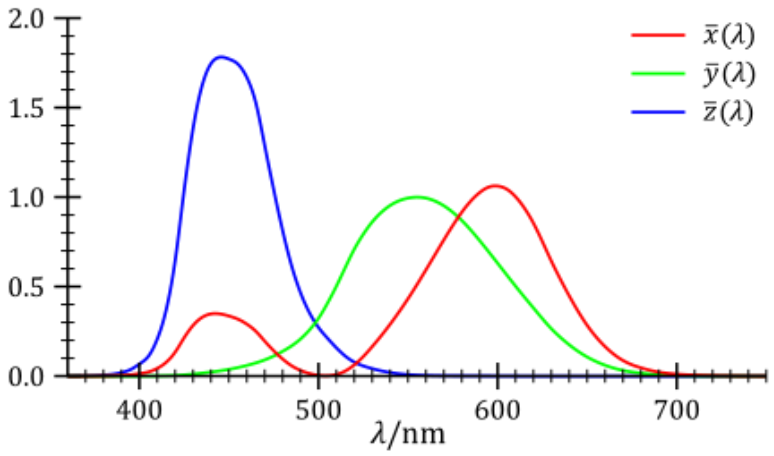

Figure 11: Human eye response curves / Tri-stimulus weighting functions

Because the $\bar{y}(\lambda)$ curve mirrors the photopic curve, the Y stimulus is actually the luminance (L_v) which gives the perceived brightness of a point in space. If instead of using the spectral radiance to compute the tri-stimulus, it was used the spectral irradiance (I_e), Y would be the illuminance measured in lux.

It is possible to transform the tri-stimulus into a color map. The CIE also created the xyY color space based on the X, Y, Z values, whose coordinates are calculated with:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{X+Y+Z} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{13}$$

The diagram depicted in figure 12 shows how the colors correlate with the above *x* and *y* coordinates:

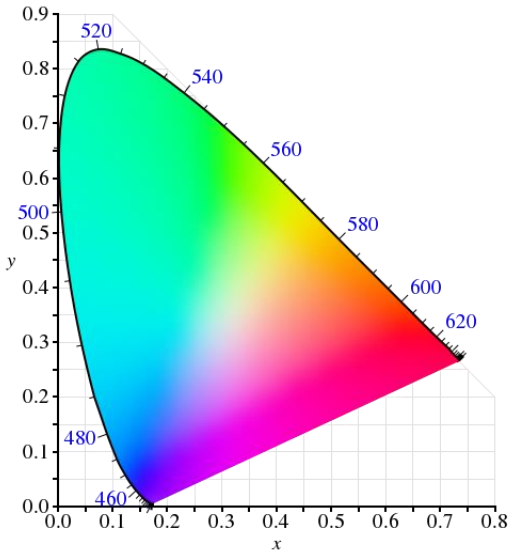

Figure 12: CIE xyY chromaticity diagram

Not all color spaces are the same. The CIE color spaces were the first but, since then, multiple new models have been designed for different purposes. Their color diagrams are usually defined by a mathematical relationship with the CIE one. The two most relevant for this work are the RGB and the YUV.

2.5.1 RGB

The RGB color model, just like the CIE, is an addictive color model with three components, where red, green and blue light components are added together to produce a wide range of colors. It is widely used nowadays in multiple devices, being the prime model featured in computers, monitors and televisions. Most cameras use and support the RGB model to allow easy integration with these devices.

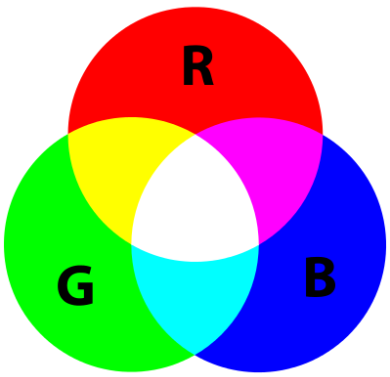

Figure 13: RGB

To obtain different colors with RGB, light beams with the color of each of its base components (red, blue and green) must be overlaid and depending on the intensity of each beam, different colors will be obtained. It is called addictive because when lights are added together, their light will add wavelengths to make the final color spectrum.

The three components may have very low intensity (typically 0 in a digital device) to max intensity (typically 255) and thus when every component is null, black is obtained, while white is the color gotten when max intensity exists in all three beams.

While there are multiple RGB color spaces, one that is widely used is the sRGB, created cooperatively by Microsoft and HP. This color space tri-stimulus can be obtained through the following calculations:

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 3.2406 & -1.5372 & -0.4986 \\ -0.9689 & 1.8758 & 0.0415 \\ 0.0557 & -0.2040 & 1.0570 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{14}$$

$$R = \begin{cases} 12.92 * R', & R' \leq 0.0031308 \\ 1.055 * R'^{\frac{1}{2.4}} - 0.055, & R' > 0.0031308 \end{cases} \tag{15}$$

$$G = \begin{cases} 12.92 * G', & G' \leq 0.0031308 \\ 1.055 * G'^{\frac{1}{2.4}} - 0.055, & G' > 0.0031308 \end{cases} \tag{16}$$

$$B = \begin{cases} 12.92 * B', & B' \leq 0.0031308 \\ 1.055 * B'^{\frac{1}{2.4}} - 0.055, & B' > 0.0031308 \end{cases} \tag{17}$$

X, Y and Z are the tri-stimulus from CIE. The resulting R, G and B values will be range from 0 to 1 but, in most of real life applications, they will be converted to be between 0 and 255 as they are usually digital stored in 8-bits. The resulting color diagram is:

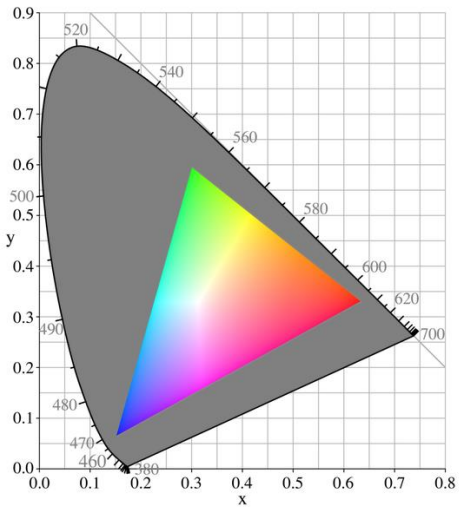

Figure 14: sRGB color diagram on top of the CIE color diagram

To do the inverse conversion (having sRGB values and converting them to X, Y, Z), a matrix multiplication is once again helpful:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4124 & 0.3576 & 0.1805 \\ 0.2126 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9505 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{18}$$

With this, it is clear that the Y value, which represents luminance, can be calculated with:

$$Y = 0.2126 * R + 0.7152 * G + 0.0722 * B \tag{19}$$

This equation, gives the perceived brightness of a pixel encoded in sRGB.

2.5.2 YUV

The YUV is a color model composed by one luma component (Y) and two chrominance components (UV). This model great advantage is that transmission and compression related errors have less impact on the image due to the encoding considering human perception so that small shifts in the values will have less impact on the user when compared to the RGB model. This property makes YUV mainly used in color image pipelines, where bandwidth is a concern and errors are to be kept at a minimum. [20]

Usually, the values of Y span from 0 to 1 while U and V from -0.5 to 0.5.

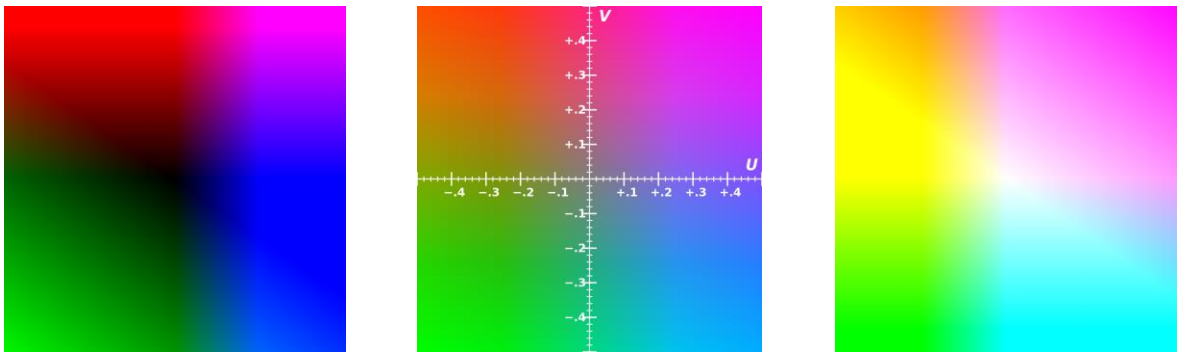

Figure 15: Color coding in YUV - From left to right: Y=0; Y=0.5; Y=1

Because one of the prime targets in this work is to be able to obtain a perception of the luminance in an area using a camera, getting the brightness in a photograph is very critical. The Y component in the YUV can provide this information upfront whereas RGB requires a formula using all three components to obtain the same result.

The chrominance components aren't as important, as specific color detection isn't a priority and color change will also be reflected in the luminance anyway.

3. Architecture and Methodology

The goal of this project is to determine whether it is possible and viable to use a single system to monitor both lighting and temperature conditions in an automated environment.

Determining lighting conditions is easy, any silicon based camera will easily capture the visible spectrum and therefore register this information. However, doing the same to the temperature is not so trivial. Two possibilities arise, either purchase a mid-infrared imager or use a regular camera without its infrared filter. Because these devices can sense wavelengths up to 1100 nanometers with a certain degree of success, we could explore what variations of temperature they detect. Of course, a microcontroller is needed to interact with the cameras and relay the information to the relevant destination, like a control station. Figure 16 illustrates a scheme of how the former option would be setup and figure 17 the latter one.

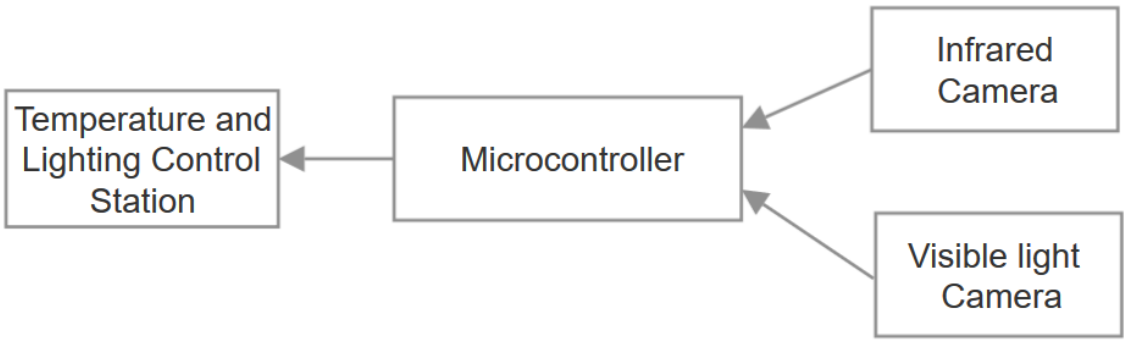

Figure 16: Option 1 diagram scheme

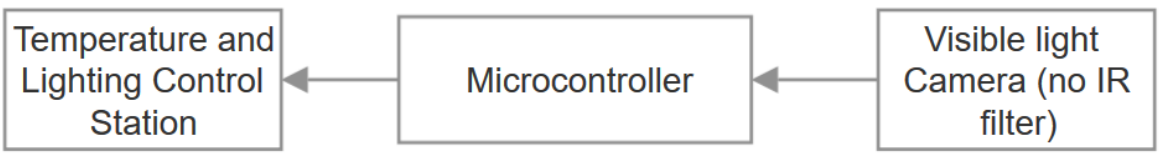

Figure 17: Option 2 diagram scheme

There are advantages and disadvantages for both options. Using two cameras will allow a more precise temperature reading, but because an extra device must be acquired the project loses a bit of adaptability since there must be room to fit it in. This option is also more expensive.

The second case will not have enough accuracy to get accurate temperature readings. At most, it can detect temperature shifts over time. The adaptability is great tough, with a single imager being able to get both lighting and temperature allow more flexibility in its placement.

In summary, for a project based upon figure 16 its most important components are a microcontroller, an infrared camera and a visible light camera. For figure 17 a microcontroller is still required, but a single visible light camera without a filter is needed. For testing purposes, in this second option, it is advised to have a way to compare filter vs no filter so for this work, to understand if there are any weird interactions.

3.1 Camera availability

Picking the right hardware for an embedded systems project can be quite difficult and this one is no exception. The most important component here is the camera used. When picking a sensor, the most relevant characteristics are its size, cost, number and size of pixels, field of view (FoV) and software/libraries available. The camera should be small to provide easier customization and integrability and as always, as cheap as possible. Obviously, it must be compatible with a microprocessor.

The field of view is the extent of the surrounding world that the camera can capture. It is measured in solid angles and is decided by the physical size of the sensor (which in turn is determined by the number and size of pixels) and the lens used. Because a fish eye lens can be attached later to provide a FoV as high as possible, this parameter is not very important in the original camera. The number and size of pixels will determine the size and resolution of the image captured. Bigger pixels will lower the resolution and increase the size of the image and more pixels just increase the size. Since in this project the objective is to find the brightness and temperature of areas in a room, low resolution is not an issue and therefore pixel characteristics is not a defining parameter.

Having good software support and libraries available is a bonus that should be considered, as it speeds up the process of learning as well as provide tools to make a successful project.

The visible light sensors that stands out the most right now in the market are the Sony IMX219 and the OV2640. [16]

Sensor	Sony IMX219	OV2640
Resolution	8 Mpixels	5 Mpixels
Color encodings available	RGB, YUV, …	RGB, YUV, …
Sensor Resolution	3280x2464	2592x1944
Sensor image área (mm)	3.68x2.76	3.67x2.73
Pixel size (um)	1.12x1.12	1.4x1.4
Horizontal FOV	63.2 degrees	
Vertical FOV	48.8 degrees	

Table 7 – Sensor specifications

The Sony sensor is clearly superior in all imaging specifications, and there are various prebuilt boards for them such as the Raspberry Pi Camera Board V2 (for the Sony sensor) featuring a CSI output or the Arducam (for the OV2640 sensor) with SPI, but both of these cameras have advantages over others for multiple reasons:

- Reliability: There are a lot of cameras with USB interfaces, but they may not work with microcontrollers. Users have compiled a list [15] of working USB webcams and the majority are faulty. The CSI and SPI outputs do not have these problems;
- Adaptability: In this work, adaptability is a key part of it. Being able to easily adapt the camera's position is important and they are small enough to allow portability. Having options like mounting different lens and the ability to choose whether they will have an infrared filter or not is something to be considered;
- Documentation: By being well established in the market, there is a broad amount of documentation available to these cameras;
- Color encodings available: Both have options regarding outputs with RGB and YUB encodings. This is a must in this work;
- Affordability: With prices around the 22€ they are not expensive.

The Sony IMX219 sensor is used in both the Raspberry Pi Camera Board V2 and the Raspberry Pi NoIR Camera Board V2. The only difference between these two cameras is the fact that the latter lacks an infrared filter, which means that it can detect near infrared, crucial to the project shown in the scheme in figure 17.

Since the aforementioned cameras use a camera serial interface (CSI) and they are made by the RPI foundation, the prime candidate for the microcontroller is the raspberry PI which is mobile, has low power consumption, is quite affordable and customizable, making it a good choice. Since the dedicated CSI port is used exclusively by the camera module produced by the RPI foundation so there are no compatibility issues here. The main downside is that there is only one CSI port per RPI, so if someone wants to use two cameras they must use either an USB camera or another microprocessor.

The market of infrared imagers is smaller than the visible light and there is not as much documentation as there are for the Raspberry PI cameras. There are few camera sensors in the market suitable for integrated projects. Lepton, for example, offers different sensors and a breakout board for easy integration such as the LEPTON Longwave Infrared Camera Module. [12]

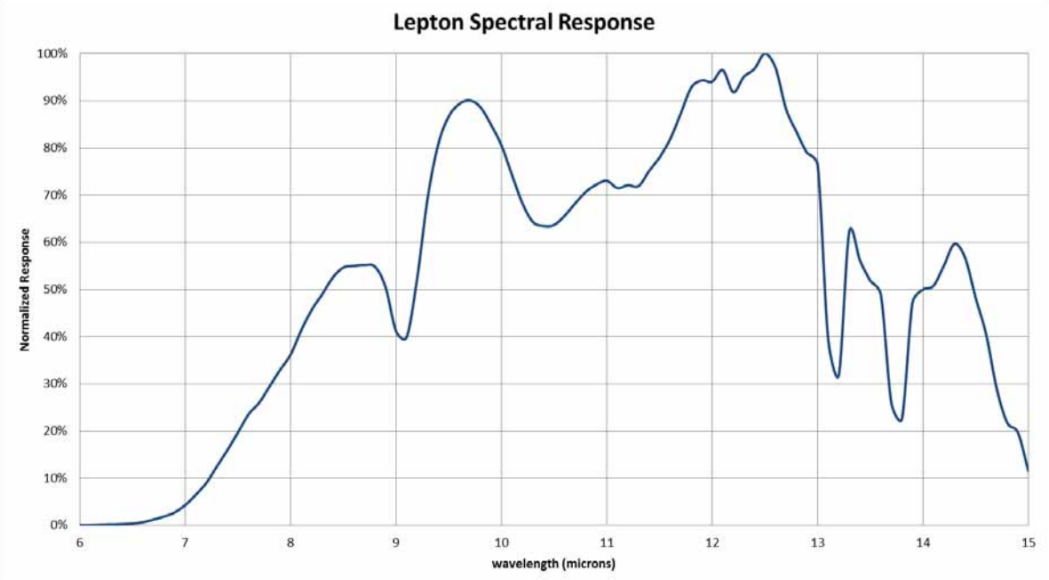

Figure 18: Lepton sensor spectral response

Unsurprisingly, these sensors have a good spectral response around the value indicated in equation 9, as it is the emitted wavelengths by ambient temperature bodies.

However, the photodetector for these optical sensors cannot be silica, which invariably leads to a higher price tag, as rarer materials are needed.

Another problem arises about the lens used in thermal imagers. Common glass has very poor transmission in the mid-infrared spectrum and other (expensive) materials must be used like germanium, since it around 50% transmission rate above 2µm. [11]

All those difficulties stack, raising the price tag of a thermal camera, even by minimizing its costs. For example, the above lepton sensor and a single breakout board can reach easily 200€, which is expensive when compared to the raspberry PI cameras.

For this reason, the setup chosen was the second one, as in the scheme in figure 17.

3.2 Camera acquisition method

The approach used in this work was based upon figure 17. This means that for the testing it was used a raspberry PI and both raspberry PI cameras. Testing with the two cameras allows a better understanding of the differences between having or not having a filter and its impact in the results.

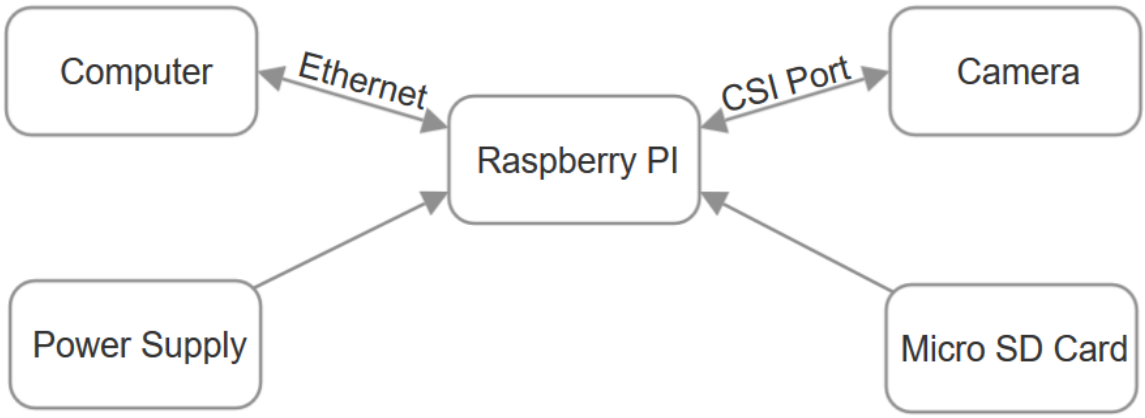

Figure 19: Simplified block diagram showing the architecture used

The data acquisition method for this project had the same base architecture for all its takes. The Raspberry PI microcontroller had an external power supply of 5V, 2.5A and a micro SD card containing the operating system Raspbian. Through its CSI port, it was connected to the camera where it gave commands and received the data captured. A remote computer, using the graphical desktop sharing system Virtual Network Computing (VNC) controlled the raspberry PI through ethernet.

The methodology used to obtain all the results in this work is based on the specifications bellow:

1. Assert the target objective for the test;
2. Create a python script that fulfill the requirements;
3. Place the camera under conditions that satisfy 1;
4. Run the script and do not interfere unless stated otherwise;
5. Collect the data stored by the microcontroller during the test;
6. Make the relevant analysis through MATLAB.

To understand this better a simple example follows:

“I want to know what is the average value of the luminance component in the YUV color model in Area A.”

The above statement is already the first step, as it describes the objective clearly. The luminance value is the Y component.

Next step is to create the python script.

```
import time
import picamera
import numpy as np
import picamera.array

#takes a picture with YUV encoding with a resolution of 512*512 pixels
with picamera.PiCamera() as camera:
    with picamera.array.PiYUVArray(camera) as stream:
        camera.resolution = (512,512)
        time.sleep(2)
        output = np.empty((512,512,3), dtype=np.uint8)
        camera.capture(stream,'yuv')
        YUV = stream.array
    camera.close()


Y = YUV[:,:,0]

#saves the Y data to an external file
file = open ("yCam.txt", "w")

for i in range(len(Y)):
    x=' '.join(str(e) for e in Y[i,:])
    file.write(x)
    file.write("\n")

file.close()
```

Figure 20: A python script that take a picture of an area and saves the luminance component to a text file called “yCam”.

Place the camera in a way that it will capture Area A and run the above script.

Transfer “yCam.txt” to a computer and run the following MATLAB script.

```
1| y=importdata('yCam.txt');
2| average_luminance = mean2(y);
```

The “average_luminance” variable should contain the solution for this example.

3.3 Methodology for ambient lighting

There are two methods to obtain the luminance perceived by the raspberry cameras. The most straightforward way is, like in the above example, to use YUV encoding and simply retrieve the Y values, which should have some relationship with the real luminance. The other method involves the RGB camera encoding. There are multiple RGB color spaces but according to the camera documentation [21], the one used is the ITU-R BT.601 version. This version is quite similar to the previously explained sRGB. A good way to compare them is through their primary colors and white point coordinates in the color space.

Colors	RED		GREEN		BLUE		White point	
Coordinates	x	y	x	y	x	y	x	y
sRGB	0.64	0.33	0.3	0.6	0.15	0.06	0.3127	0.3290
ITU-R BT.601	0.64	0.33	0.29	0.6	0.15	0.06	0.3127	0.3290

Table 8 – Color coordinates in ITU-R BT.601 and sRGB

Because they are nearly identical, equations 14, 15, 16 and 17 for calculating sRGB color spaces can be used with this variant.

However, BT.601 has a different conversion matrix to YUV.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{20}$$

Therefore, the actual perceived luminance of this RGB variant is:

$$RGB\ Luminance = 0.299 * R + 0.587 * G + 0.114 * B \tag{21}$$

3.4 Temperature readings from a camera

The NoIR camera detects the near-infrared spectrum. In chapter 2.2 it was shown that a black body at 27ºC (300Kelvin) will emit peak radiation at 9.66 μm. It is a long way from the mere 1.1 μm the camera can detect and with a poor quantum efficiency. With help from the Wien displacement law, it is known that a wavelength peak 1.1μm occurs at 2634K. This means that the camera will efficiently detect temperature changes at least around this value. However, very few things can reach this amount. A very hot day on earth is 40ºC and a common oven will easily reach 200ºC so if the camera could detect variations around these temperatures it would be more interesting.

The black body radiation spectrum is defined by Planck's law given by:

$$L_e(\lambda) = \frac{2hc^2}{\lambda^5}\left[e^{\frac{hc}{\lambda kT}} - 1\right]^{-1} \tag{22}$$

In this equation, h is Planck's constant, c is the speed of light, k is the Boltzmann's constant, T the temperature in Kelvin degrees and λ the wavelength. With this formula it can be computed the radiation emitted by a body at near-infrared wavelengths at ambient temperature. At 20ºC, (293K) 30ºC (303K) and 40ºC (313K):

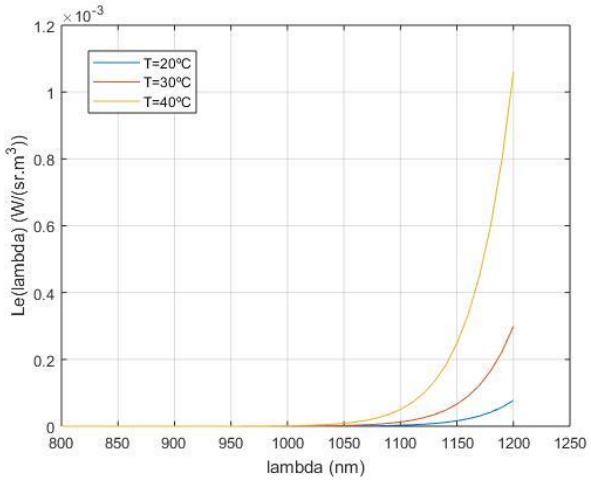

Figure 21: Radiation emitted by a black body at 20, 30 and 40ºC.

The amount of energy emitted at these temperatures are minimal. Supposing that the camera will absorb at 100% efficiency the wavelengths from 800 to 1100 nm, the total power captured will be:

$$P_{20ºC} = \int_{800nm}^{1100nm} L_e(\lambda)\, d\lambda = 76.9 * 10^{-5}\, W/sr/m^2 \tag{23}$$

$$P_{30ºC} = \int_{800nm}^{1100nm} L_e(\lambda)\, d\lambda = 344 * 10^{-6}\, W/sr/m^2 \tag{24}$$

$$P_{40^{\circ}C} = \int_{800nm}^{1100nm} L_e(\lambda)\, d\lambda = 1427 * 10^{-6}\ W/sr/m^2 \tag{25}$$

A common office usually needs 300 lux. A candela is equal to 1/683 W/sr at 555nm [3]. If we suppose that a monochromatic wave at 555 nm is being used to illuminate said office, then the radiant power of this beam is:

$$P = \frac{300}{683} = 0.4392\ W/sr/m^2 \tag{26}$$

Bear in mind that this is the best-case scenario. If multiple wavelengths were in place, they would have to be scaled up accordingly through the photopic curve. It is unlikely that a near infrared camera would be able to detect temperature shifts at ambient temperature. With the above example, at 30ºC, only $7.83*10^{-4}$% of the total radiance is from the black body temperature emission. A camera has 8 bits to store RGB/YUV values. A change in the environment would have to be at least $1/256=3.9*10^{-3}$% for a camera to detect it.

$$P_{temp}\% = \frac{P_{temp}}{P_{temp} + P} \tag{27}$$

Therefore, at the best-case scenario, to obtain any effect with a NIR camera, the radiation captured from temperature (P_{temp}) must be at least $1.72 * 10^{-3} W/sr/m^2$.

This value is obtained at 42ºC, using the same 800 to 1100nm interval. Because these calculations used so many approximations, like supposing the emission was coming from a black-body or that all the non-temperature radiance was from a 555nm beam, it is very unlikely that a perfect camera with no noise would be able to detect anything at this temperature.

4. Testing camera characteristics

4.1 Color encoding in the Pi Camera

The Raspberry Pi camera offers two possibilities regarding data storage. YUV and RGB encoding are both supported and available as means to capture information. It was previously established that luminance is a very important metric and therefore YUV has the upper hand in this case. However how different are they? Should the need of using RGB for some reason arise, will the conversion significantly alter the data? To settle this matter, the python script was used to capture the area shown in figure 22.

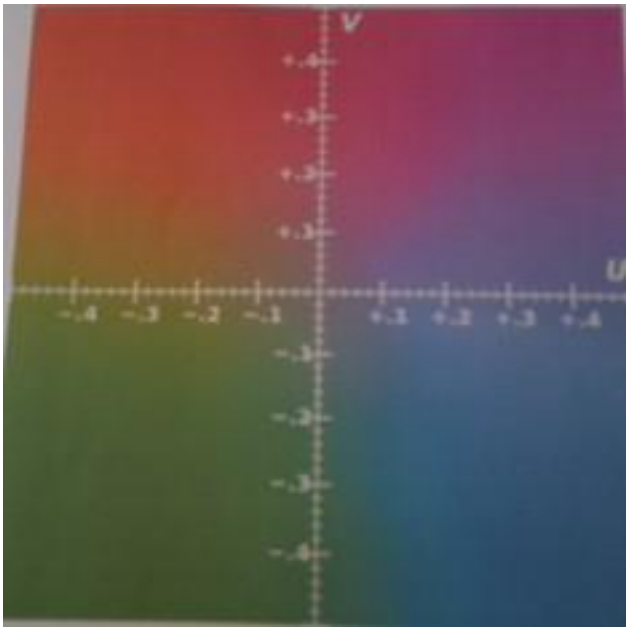

Figure 22 – Picture used in the current test

Choosing a colorful scene is important for this test case. In a single image most of the spectrum can be analyzed, and deviations are easier to find.

By using the script, three sets of data were obtained. The first is the YUV, the second a RGB conversion of the previous YUV information and finally there is the standard RGB data. They were all decomposed in their respective components. For example, the YUV set is composed by the matrixes of its elements Y, U and V separately. All matrixes size is 256x256 which is the picture's pixel count. The values can, theoretically, range from 0 to 255.

First let us take a look on the perceived brightness of the YUV and RGB data. For YUV it is easy, the luminance Y gives directly this knowledge (1). For RGB, equation 21 must be used (2). The maximum number obtained in both cases was 163 and the minimum was 60. After subtracting (1) to (2) and rounding the values to the nearest integer table 1 in appendix B is obtained. Histogram 1 is shown below for easy readings.

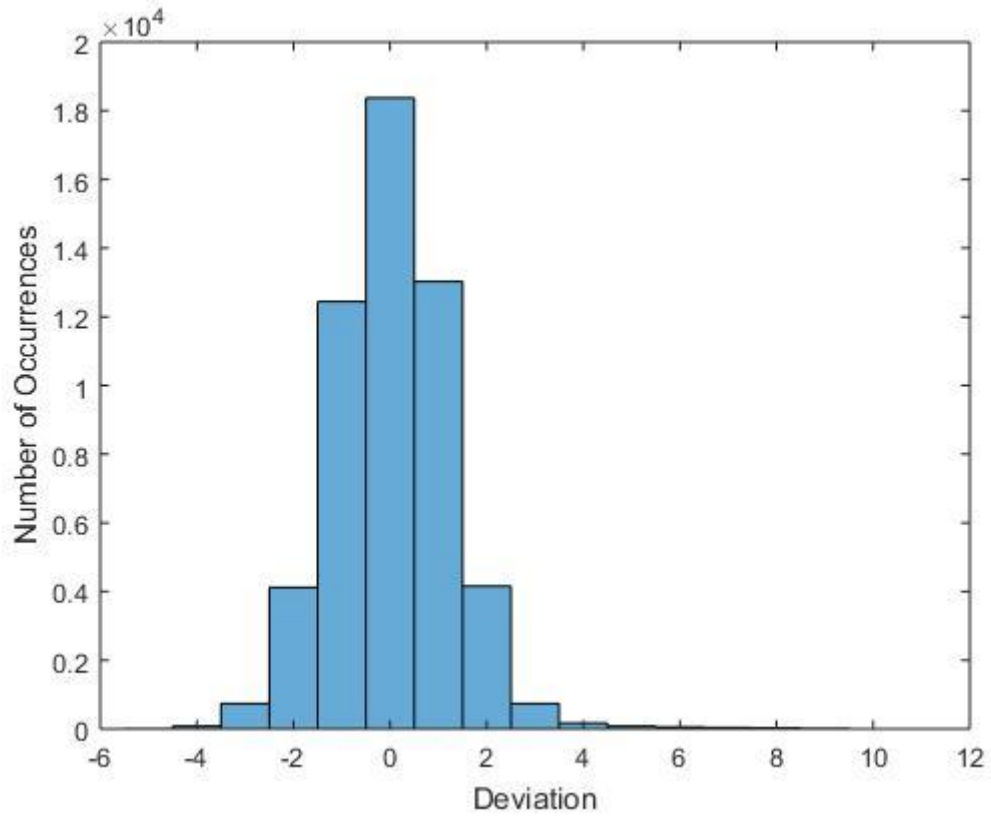

Figure 23: Difference in brightness measured with RGB and YUV

As it turns out, the average value of the deviation between YUV and RGB brightness, in this scenario, is 0.0446, and has a standard deviation of 1.2572. Since the data range is 103, it can be concluded that there is little difference in checking the perceived brightness by one model or the other.

Comparing the differences with both luminance and chrominance components is a little trickier. Complex formulas have to be used to make an effective conversion and obtain the aforementioned second set of data, the converted YUV. By comparing them in the same fashion as the above case the following histograms are obtained.

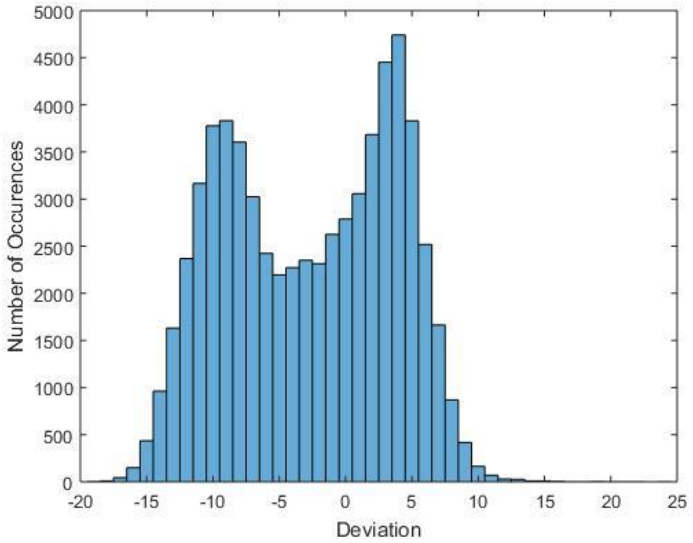

Figure 24: Difference between the R component in RGB and R converted from YUV

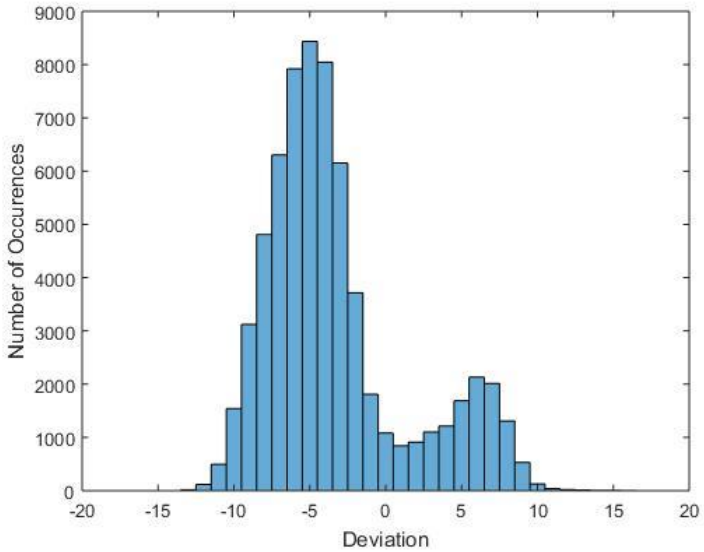

Figure 25: Difference between the G component in RGB and G converted from YUV

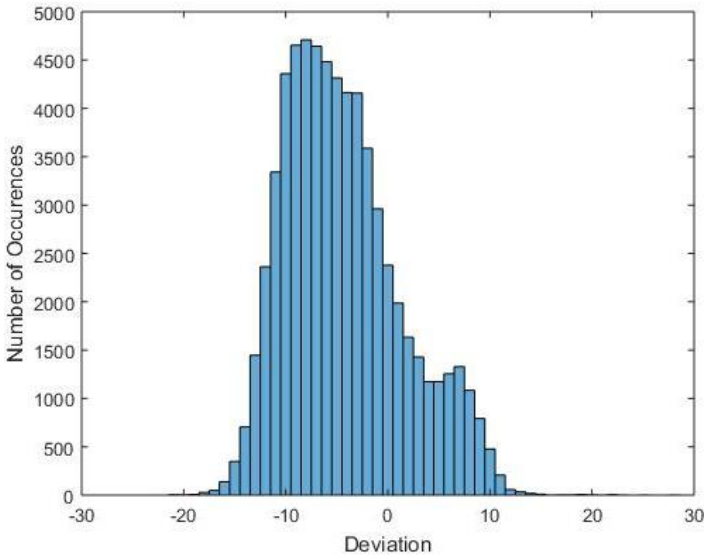

Figure 26: Difference between the B component in RGB and B converted from YUV

Component	R	G	B
Average	-2.6849	-3.3429	-4.3374
Standard deviation	6.3968	4.7402	5.7779

Table 9: Average and standard deviation of the difference between an RGB encoding and a YUV encoding converted to RGB

Component	R	G	B	R converted	G converted	B converted
Maximum	199	169	172	210	177	181
Minimum	37	43	35	23	34	22
Range	162	126	137	187	143	159

Table 10: Maximum, minimum and range of the values obtained in RGB encoding and a YUV encoding converted to RGB

These results show that the difference may have some impact when analyzing results across different models. However, what really matters is the perception humans have regarding color. A difference with theses magnitudes may not cause noticeable shifts in color.

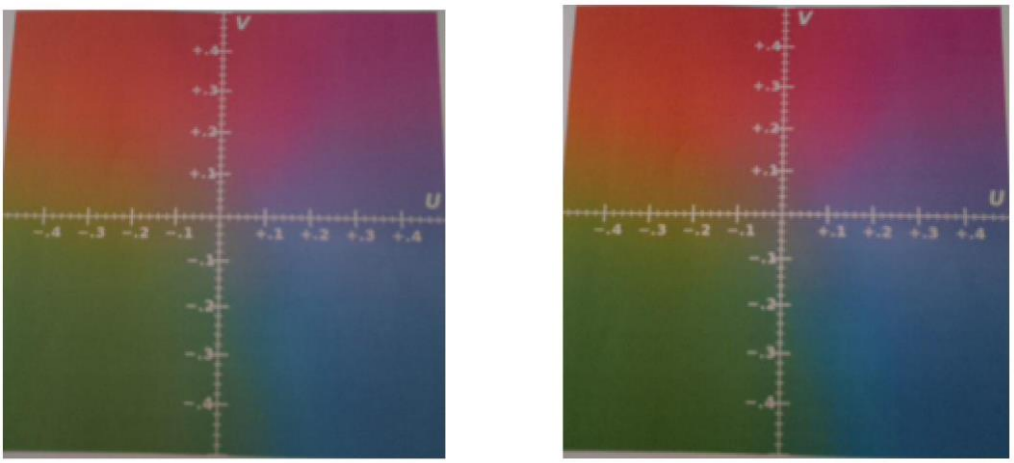

Figure 27 and 28: RGB encoding and YUV encoding

Figures 27 and 28 shows the same picture but the left one is directly encoded in RGB while the right one was encoded in YUV and converted to RGB afterwards. A keen eye may observe that they are slightly different, but nothing groundbreaking.

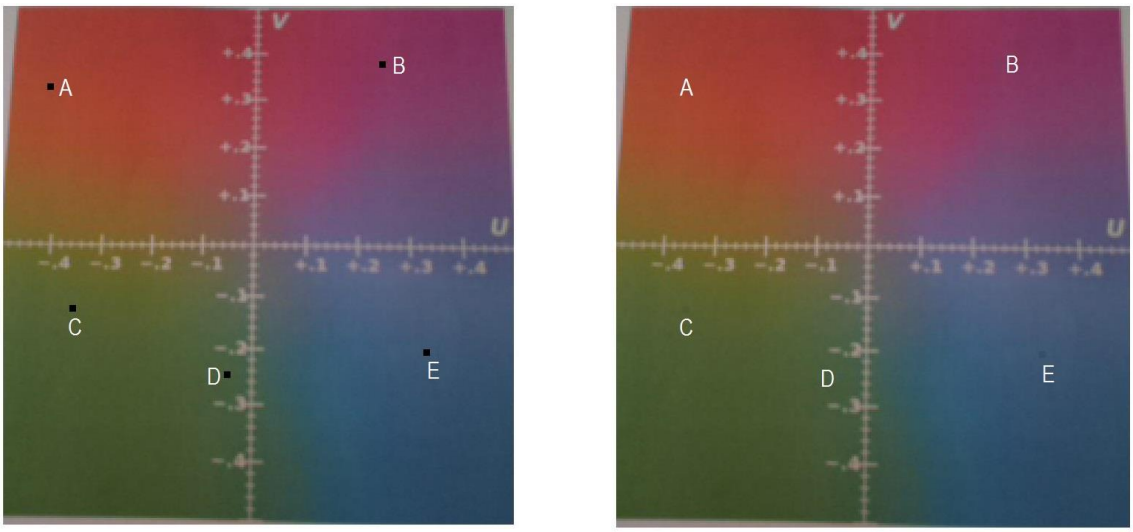

Figure 29 and 30: Left image shows figure 23 with 5 points set to 0. Right image substitutes these points with data from figure 24

By taking figure 27 and substitute some pixels taken from figure 28, it can easily be seen that there is indeed a difference and conversions may be more accurate depending on the color. For the curious one, check table 15 in appendix B to see the average value of the substituted pixels.

It could be explored further, but there is no need to do so. In summary, both RGB and YUV are viable to analyze any given picture, with a slight advantage to YUV as it directly specifies perceived brightness. Conversions between both models are possible but should be avoided. Even though there are few differences, they exist and for this reason, once a model is chosen, you should stick with it to prevent inconsistencies.

4.2 Photometry on cameras

With the analysis of the encoding done, it is now interesting to know whether is possible to establish a relationship between photometric units and the color spaces obtained from the cameras. Figure 31 was used to help solve this.

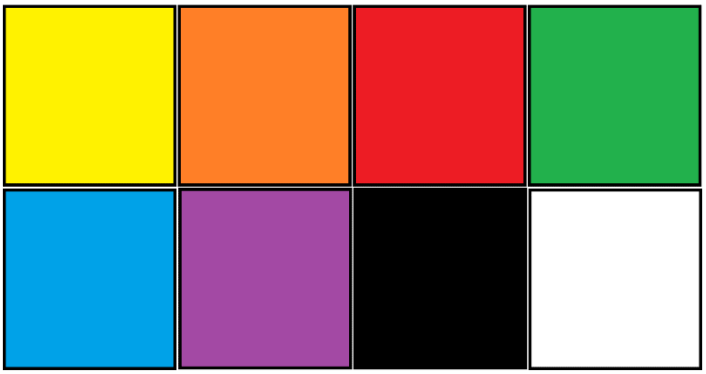

Figure 31: Image used in this chapter

It provides a broad amount of test cases with colors all around the spectrum, which hopefully will allow to get a good feeling of how will photometric units vary according to different colors and light intensity. The conditions and procedure used in this test were:

1. Set a computer screen with figure 31 shown at max brightness;
2. Through a smartphone lux meter, measure each individual color lux;
3. Capture a snapshot of the computer screen using both the normal Raspberry Pi camera and the NoIR Raspberry Pi camera (YUV encoding was used);
4. Lower significantly the brightness of the screen and repeat steps 2 and 3.
5. By using the illuminance of the white square as control case, calculate every color's ratio.

The results of the lux meter measures are in the table 11:

Color	Maximum brightness on screen		Lower brightness on screen	
	Illuminance (lx)	Ratio to white	Illuminance (lx)	Ratio to white
Yellow	124	67 %	24	69 %
Orange	90	49 %	17	49 %
Red	60	33 %	11	31 %
Green	27	15 %	5	14 %
Blue	69	38 %	13	37 %
Purple	61	33 %	12	34 %
Black	0	0 %	0	0 %
White	183	100 %	35	100 %

Table 11: Lux meter results

Color	Maximum brightness on screen		Lower brightness on screen	
	Normal camera	NoIR camera	Normal camera	NoIR camera
Yellow	141 (80%)	125 (76%)	146 (81%)	122 (76%)
Orange	82 (47%)	101 (61%)	78 (43%)	97 (61%)
Red	61 (35%)	70 (42%)	60 (33%)	69 (43%)
Green	44 (25%)	30 (18%)	53 (29%)	34 (21%)
Blue	79 (54%)	78 (47%)	85 (47%)	72 (45%)
Purple	79 (34%)	96 (58%)	78 (43%)	92 (58%)
Black	15 (9%)	16 (10%)	14 (8%)	17 (11%)
White	176	165	180	159

Table 12: Y value results

Unsurprisingly, different brightness levels do not influence the ratio. There is a sharp drop in illuminance, but it was expected since less light power is being emitted by the screen.

By observing the results in table 12, it can be seen that both cameras yielded similar results. Even though ratios are quite different from one camera to another, it is expected as the infra-red camera will always lean towards the red color as it lacks the filter. Altering the brightness does not change the values of any components, because of the auto-exposure camera feature that automatically adapts the camera to the environment. This means that the cameras will be useful in detecting different light levels in a wide area. If a room is divided by sectors and a sensor is used to monitor all of them at once, it would be easy to control the lighting in these divisions.

Unfortunately, it seems that a connection between photometric units and color models is not possible with these conditions. Changing the brightness of an area causes the illuminance to drop, but the camera does not detect any noticeable changes. This occurs because of the auto-exposure time feature present in cameras. In order to capture a scene as visible as possible, cameras control the exposure time automatically based on the maximum value of a section. Should exist a way to control the exposure time manually, the detection of the aforementioned changes will be possible.

4.3 The NoIR camera and temperature

The setup in this test was as follows:

1. Heat a heat rock to a temperature as high as possible (it was 200ºC (473 Kelvin) in this case);
2. Place the heat rock as shown in figure 32;
3. Take pictures with the raspberry pi NoIR camera and measure the corresponding temperature;
4. Keep doing 3 until the heat rock is at ambient temperature.

Figure 32: Scene used in this test

Afterwards, by using the script shown in appendix F the data gathered was split between the rock and the rest of the scene and average values for both sets were calculated. In theory, since only the rock's temperature will decrease over time and lighting shifts will affect both the rock and the scene, if the temperature drop has influence over the camera's captures, then the rock's data will change, and the rest of the scene will not. Bear in mind that even with the exact same conditions, shifts of 1/2 are expected since the camera is not 100% reliable.

Temperature (ºC)	Average value of heat rock			Average value of rest of scene		
	Y	U	V	Y	U	V
200	67	127	139	103	132	143
180	67	125	140	101	130	144
160	66	125	139	100	129	144
140	65	125	140	101	130	145
120	67	124	140	101	129	144
100	67	125	141	101	129	144
80	66	125	140	101	130	145
60	65	124	140	101	129	145

50	66	124	141	101	128	145

Table 13: YUV values vs temperature

From chapter 3.4, it was already known that under ideal conditions, no changes would occur below 42ºC, so it is pointless measuring anything below that. The results show that, at least in this temperature range, the camera cannot detect changes based in temperature shifts. None of the three components changed by a number larger than 2, over the time period, which is the error margin of the camera. Sometimes they drop just to raise in the next measurement, so the final conclusion is that near-infrared cameras won't be able to detect temperature shifts under 200ºC.

5. Camera testing in a real-world environment

5.1 Equally lighted room

The best way to understand how a camera would work in a real scenario is to use a room with a controlled environment (no external disruptions, expect for the change of weather). The room must be monitored during a full day from dawn to dusk with an equalized source of natural light (a window that spans the whole room for example) and it should have some normal clutter (desks, cabinets, etc). To fill in the criteria, the room shown in figure 33 was used. The python script used can be seen in appendix A to obtain YUV information. The day in question was a summer sunny day with no noticeable clouds and the data was collected from 5a.m. to 9p.m. in intervals of 5 minutes to capture all the light moments.

Figure 33: Scene used

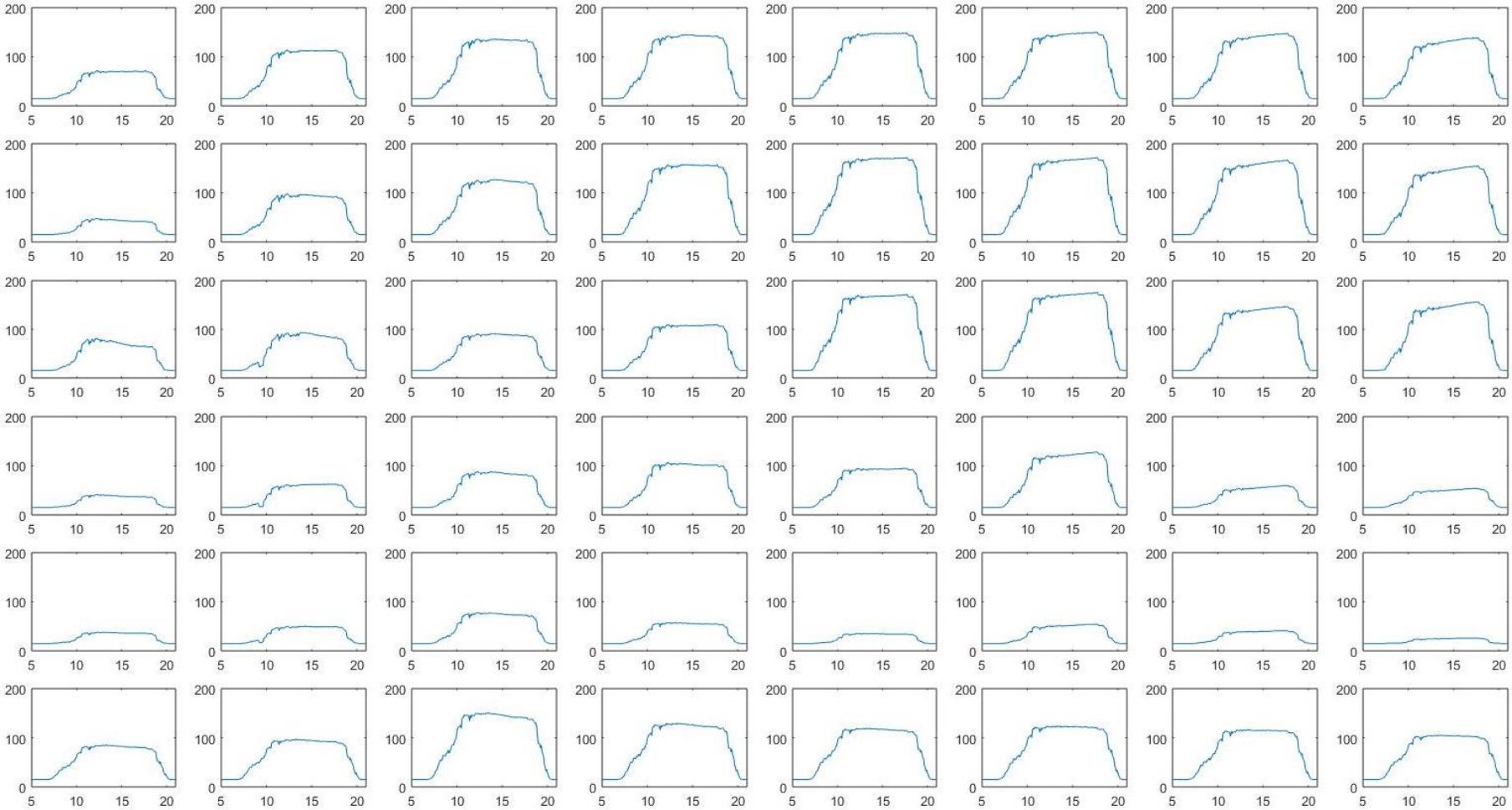

Figure 30: Y data collected divided in 6*8 sections

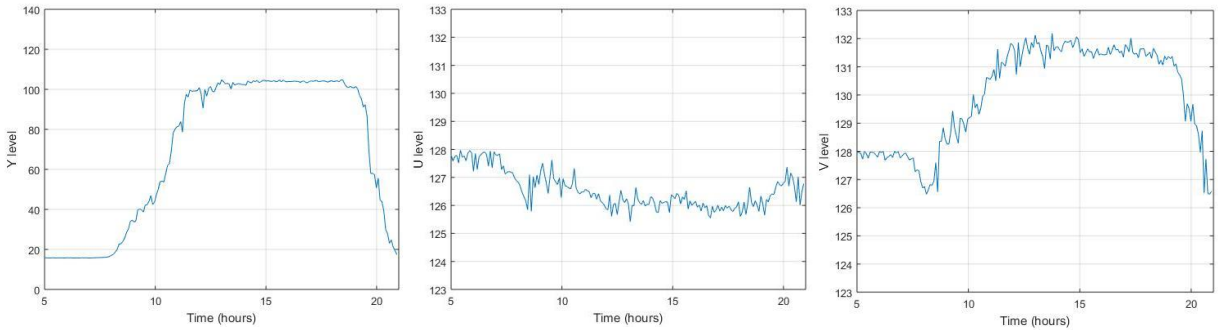

Figure 34: From left to right: average Y, U and V values of the scene from 5a.m. to 9p.m.

Figure 34 shows the average of all components during the data collection period over the whole area. One of the things that stands out is the messy graphics of the chrominance components. Somewhat understandable since multiple colors are present in the area scanned, and they react differently to light changes. Anyway, this is not very relevant since color detection is not a priority.

The luminance graph is very interesting though. It shows that when no light is present, Y is set to a minimum of 17 and as the sun rises and more light enters the room the luminance obviously grows.

However, during the day the luminance does not change significantly and stabilizes until the night kicks in where it drops sharply.

Diving the area in different sub-divisions is more revealing. As it can be observed in figure 30, even though everything has equal access to the light source, some places will yield less luminance than others due to its color. This means that even if the light level across all sub-divisions is exactly the same, the sensor will detect it in a different way. Thus, the only solution to discover the relationship between optimal light level and luminance Y, is to perform a control measurement in the required conditions.

Should these sections have individual artificial light sources, they could be activated when the corresponding Y level is lower than the threshold defined by the control measurement. This way, automated environments should have no problem controlling light levels with the assistance of this camera sensor.

5.2 Unequally lighted room

Like in the previous chapter, a room will be analyzed from 5a.m. to 9p.m. in intervals of 5 minutes on a sunny day. However, in this test the natural light source (window) exists in only one side of the test area, which will lead to an unequal lighting across the room.

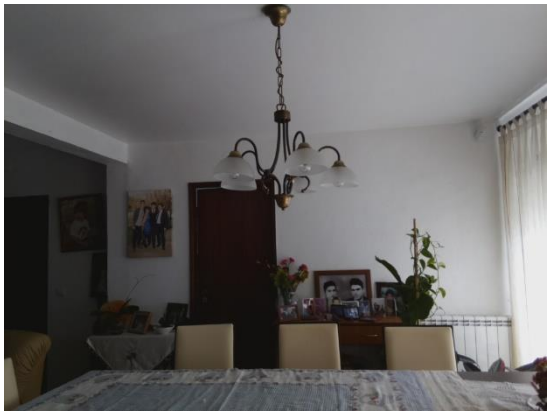

Figure 35: Room used in this test

In this test, both cameras will be used.

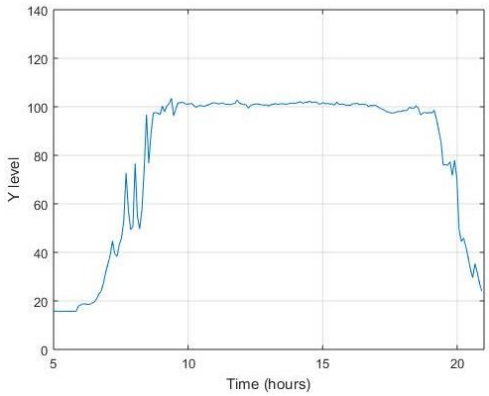

Figure 36: Average Y values of the scene shown in figure 18 from 5a.m. to 9p.m. Taken with the normal camera

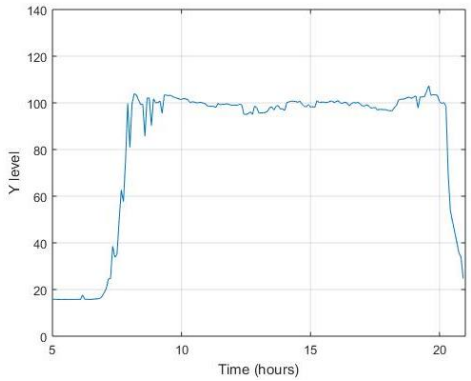

Figure 37: Average Y values of the scene shown in figure 18 from 5a.m. to 9p.m. Taken with the NoIR camera

Just like in the previous chapter, chrominance values are nearly worthless with little information to be extracted. The luminance component however states a very interesting scenario. The NoIR reaches its peak luminance value way sooner than the regular camera. Also, when dusk arrives, both reach the bottom value at the same time, but the NoIR starts its descent later. Because there is no filter, the NoIR will capture both visible and infra-red photons, which allows it to reach its maximum earlier and sustain the it to a later time. This is a problem because even though the NoIR detects that the light level is still at maximum, that's not true because humans cannot detect those extra little photons above 750 nanometers.

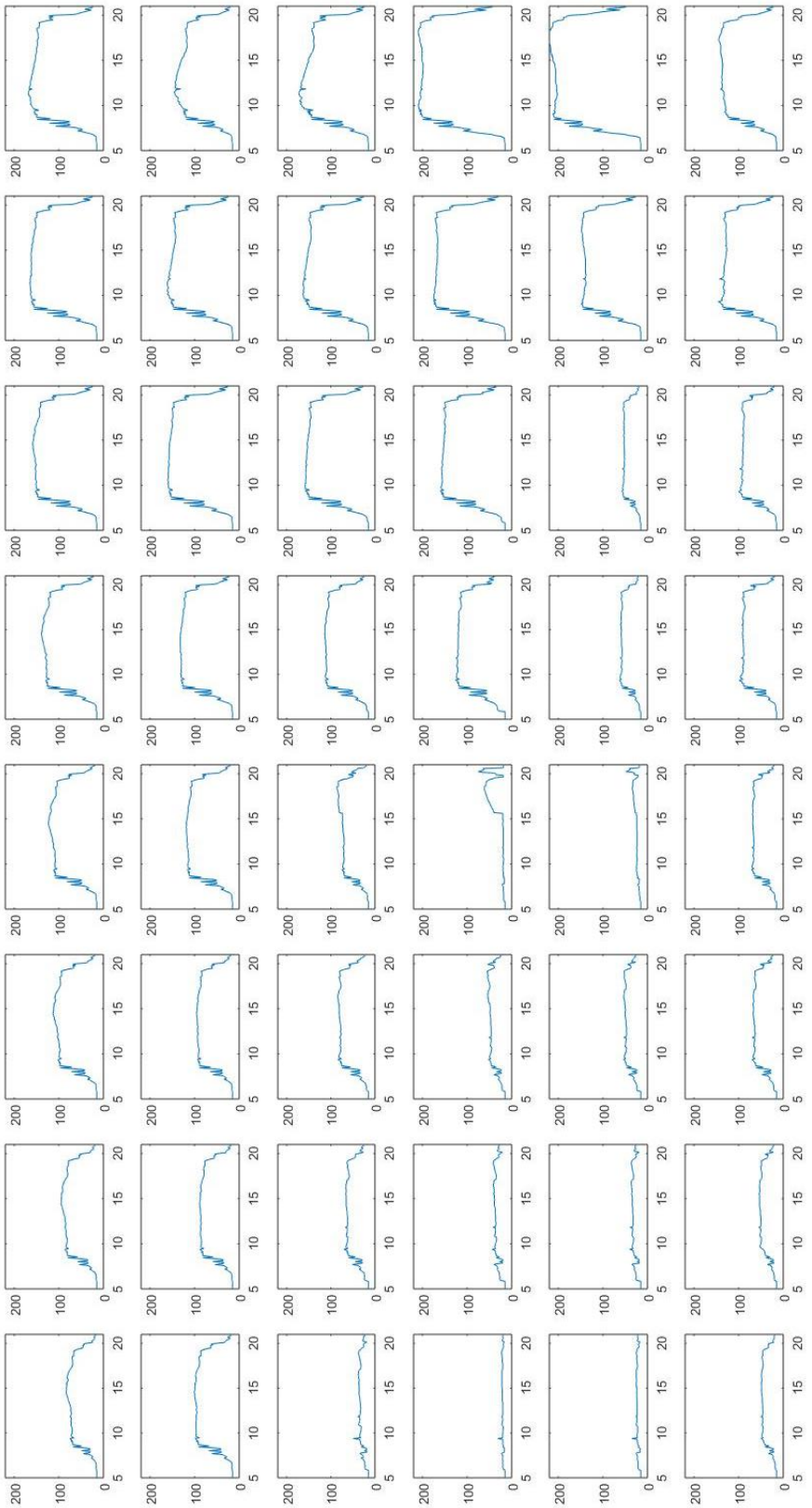

Figure 38: Y data collected divided in 6*8 sections, normal camera

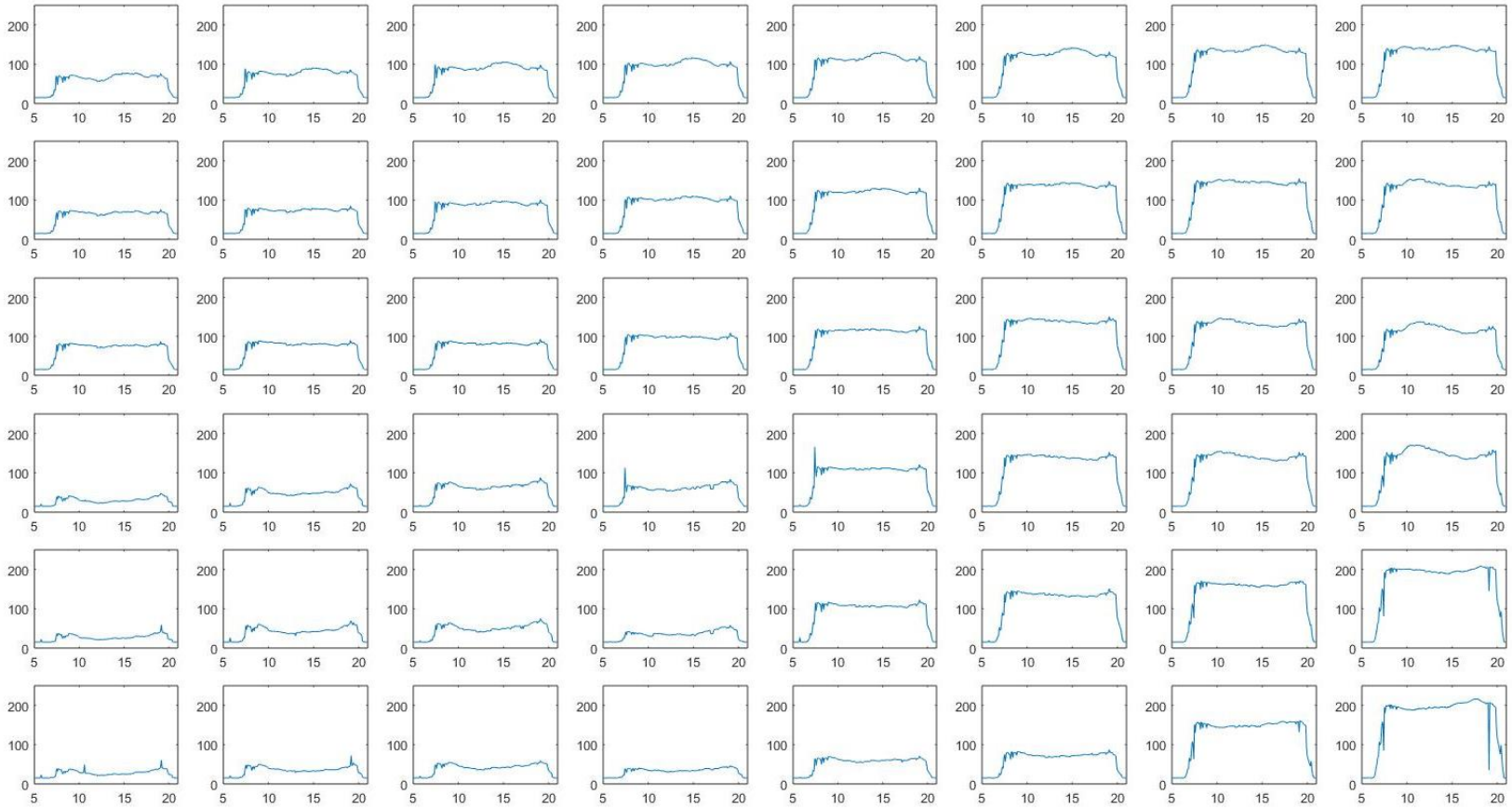

Figure 39: Y data collected divided in 6*8 sections, NoIR camera

Now, onwards to the analysis of the sectioned brightness of this room. Both results are very similar, the confirmation that the lack of filter in the NoIR camera will make it go up sooner and down later. Interestingly, the values across sectors are pretty similar. It seems that even though the NoIR is being bombarded with larger amounts of photons during peak times, the absolute Y value isn't greater. This has to do with the cap that cameras seem to have, and they adjust automatically to the highest value onscreen. Recapping chapter 4.2 where it states that to an extent, higher/lower luminance levels will not alter the encoding absolute values significantly as long as the ratios between the highest value captured and any other values remains the same. When in an automated environment, with an equal amount of artificial lighting in the room, should one of them malfunction, either by going too bright or simply stops working, by comparing it to the standard norm in the other sectors, the camera will detect it is malfunctioning and be able to adjust the light accordingly.

6. Conclusions and Future Work

6.1 Conclusions

This work tried to explore the concept of a light and temperature sensor based on a single camera device. The cameras used were silicon based, which meant they could detect both visible and the near-infrared spectrum. The two analyzed color spaces, RGB and YUV, are viable options for light level mapping. YUV has a slight advantage due to being able to access this information immediately, while RGB will require the usage of a simple conversion. There is not a significant difference between using either color space, but it is unadvised to make constant conversion to one another as approximation errors will pile up.

Because the cameras have an auto-exposure feature, establishing a relationship between photometric units and its color spaces is not possible. They seem to have a "standard" in how much photons it will capture in one take and will try to obtain an "acceptable" luminance value before closing the shutter. Unless the time exposure is fixed and known, any attempt to correlate photometry and color spaces will not be successful since their relationship will change with any slight variation in the environment.

Temperature readings at low temperatures is impossible. The slight quantum efficiency in the near-infrared spectrum of the silicon cameras is not enough to determine temperature shifts. There is some potential with higher temperatures though. A cooling feature in a camera, for example, will decrease its inherent noise, therefore increasing the dynamic range making it better at detecting small changes in radiation received.

A camera sensor can be of great use in the monitor of lighting conditions in a wide area with plenty of functionalities to obtain an efficient control of an automated environment. Cameras without an infrared filter for lighting control are not advised as the extra photons from infrared are worthless for human visibility but the infrared camera will detect and factor them in anyway.

6.2 Future Work

To explore the architecture used in this work further, the main pre-requisite is acquiring a camera that allows manual tuning of the exposure time. Theoretically, it should be possible to connect photometric quantities to the camera color space at a specific exposure time, since it can be controlled the number of photons entering the lens. By using this information, it is possible to accurately measure luminance in a room using a single device and in low light level scenarios, increasing the exposure time will allow more photons to enter thus allowing better responsivity in this situation. The manual tuning will also help when surveilling an area. Instead of being able to only detect hard drops in light levels, small fluctuations can be found as well.

Regarding temperature readings with a NIR camera, only one with a very good SNR ratio could have any chance at seeing temperatures changes. Small variations in the noise could mean changes in the readings, even with the same conditions, as temperature related radiation is very small. For ambient temperature, it is impossible to detect anything, so the only option is to use a mid-infrared camera.

The best solution for an integrated sensor that will fill in all the pre-requisites is a normal silicon camera with an infrared filter for light control and a mid-range infrared camera for temperature. Since the latter kind of cameras tend to be very expensive, this solution would require a larger budget to able to create this new device, but it will obtain very good results. The visible light camera, with an option of manual tuning of exposure time, will be able to measure luminance and an infrared camera that can accurately measure temperature could be an expensive but efficient device worth exploring.

7. References

[1] M. Weiser, R. Gold, J. S. Brown, The origins of Ubiquitous computing research at PARC in the late 1980s, accessed at (http://www.cs.cmu.edu/~jasonh/courses/ubicomp-sp2007/papers/03-weiser-origins.pdf) in 28/08/2017

[2] Ian Ashdown, P. Eng, LC, FIES, "Photometry and Radiometry - A Tour Guide for Computer Graphics Enthusiasts", byHeart Consultants Limited

[3] Webpage "Base unit definitions: Candela" accessed at (http://physics.nist.gov/cuu/Units/candela.html)

[4] By Dicklyon at English Wikipedia - Transferred from en.wikipedia to Commons by Ashishbhatnagar72., Public Domain, accessed at (https://commons.wikimedia.org/w/index.php?curid=4124751)

[5] Webpage "How to convert lumens to watts", accessed at (http://www.rapidtables.com/calc/light/how-lumen-to-watt.html)

[6] Webpage "Deciding LED Light Quantity – How many is much?", accessed at (http://www.charlstonlights.com/blog/led-light-quantity-how-many-much)

[7] Webpage "Optical Properties of Silicon", accessed at (http://www.pveducation.org/pvcdrom/materials/optical-properties-of-silicon)

[8] Dave Litwiller, "CCD vs. CMOS: facts and fiction"

[9] Webpage "Photoelectric effect (article) | Photons | Khan academy", accessed at (https://www.khanacademy.org/science/physics/quantum-physics/photons/a/photoelectric-effect)

[10] Webpage "Photoelectric effect", accessed at (https://en.wikipedia.org/wiki/Photoelectric_effect)

[11] Webpage "About sight glass", accessed at (https://www.encole.com/articles/about-sight-glass/)

[12] FLIR LEPTON® Long Wave Infrared (LWIR) Datasheet

[13] Sony IMX219PQH5-C image sensor datasheet

[14] Luis Nero Alves, Luis Rodrigues, José Luis Cura, VLC Theory and Applications – Chapter 2: Lighting and Communications: Devices, Systems

[15] RPI USB Webcams, accessed at (https://elinux.org/RPi_USB_Webcams)

[16] Webpage "Camera module – Raspberry PI documentation", accessed at (https://www.raspberrypi.org/documentation/hardware/camera/)

[17] Webpage "Wien's displacement Law - Wikipedia", accessed at (https://en.wikipedia.org/wiki/Wien%27s_displacement_law)

[18] Dany Pascale, "A review of RGB color spaces ...from xyY to R'G'B'", Babel Color, 2003

[19] Webpage "CIE 1931 color space - Wikipedia", accessed at (https://en.wikipedia.org/wiki/CIE_1931_color_space#CIE_RGB_color_space)

[20] Webpage "YUV - Wikipedia", accessed at (https://en.wikipedia.org/wiki/YUV)

[21] Webpage "picamera – Picamera 1.13 Documentation", accessed at (http://picamera.readthedocs.io/en/release-1.13/)

[22] Webpage "Infrared - Wikipedia", accessed at (https://en.wikipedia.org/wiki/Infrared)

[23] Webpage "What is a CCD? – charge coupled device", accessed at (http://www.specinst.com/What_Is_A_CCD.html)

[24] Akikazu Sakudo, “Near-infrared spectroscopy for medical applications: Current status and future perspectives”
[25] Webpage “What’s that blue thing doing here?”, accessed at (https://www.raspberrypi.org/blog/whats-that-blue-thing-doing-here/)

I. Appendix A

The following program was written in the Python language with the objective of controlling the Pi camera through the raspberry Pi. Its purpose is to take pictures of a room during a day in intervals of 5 minutes and saving the YUV picture data in a text file for a Matlab analysis.

```
import time
import picamera
import numpy as np
import picamera.array

width = 768
height = 1024

inicio = 1;
fim = 200;
sono = 255

time.sleep(22000)

#Take 200 Pictures
for z in range(inicio, fim):

    dia = str(time.localtime(time.time()).tm_mday) + "_" +str(time.localtime(time.time()).tm_mon)
    tempo = str(time.localtime(time.time()).tm_hour) + "_" +str(time.localtime(time.time()).tm_min) + "_" +
    str(time.localtime(time.time()).tm_sec)

    #Take picture
    with picamera.PiCamera() as camera:
        with picamera.array.PiYUVArray(camera) as stream:
            camera.resolution = (1024,768)
```

```
        time.sleep(2)
        output = np.empty((1024,768,3), dtype=np.uint8)
        camera.capture(stream,'yuv')
        YUV = stream.array
    time.sleep(2)
    nome = "foto_"+tempo+".jpg"
    camera.capture(nome)
    camera.close()

# Dump YUV data in a textfile
Y = YUV[:,:,0]
nome = "y_"+tempo+".txt"
file = open (nome, "w")

for i in range(len(Y)):
    x=' '.join(str(e) for e in Y[i,:])
    file.write(x)
    file.write("\n")
file.write("\n\n\n")

file.close()


U = YUV[:,:,1]

nome = "u_"+tempo+".txt"
file = open (nome, "w")

for i in range(len(Y)):
    x=' '.join(str(e) for e in U[i,:])
    file.write(x)
    file.write("\n")
file.write("\n\n\n")

file.close()


V = YUV[:,:,2]
```

```
nome = "v_"+tempo+".txt"
file = open (nome, "w")

for i in range(len(V)):
    x=' '.join(str(e) for e in V[i,:])
    file.write(x)
    file.write("\n")
file.write("\n\n\n")

file.close()

#create a file with the .txt file names to allow easy integration with matlab
file = open ("matlove", "a")
nome="y(:,:," + str(z) +")=importdata('y_"+tempo+".txt');\n"
file.write(nome)
nome="u(:,:," + str(z) +")=importdata('u_"+tempo+".txt');\n"
file.write(nome)
nome="v(:,:," + str(z) +")=importdata('v_"+tempo+".txt');\n\n"
file.write(nome)
file.close()


print(z)
time.sleep(sono)
```

II. Appendix B

Deviation	Number of occurrences
-5	7
-4	79
-3	733
-2	4117
-1	12441
0	18374
1	13026
2	4152
3	733
4	169
5	81
6	52
7	36
8	31
9	18
10	2
11	3

Table 14 – Difference between brightness measured with RGB and YUV

Component	R	G	B	R converted	G converted	B converted
Point A average	163	70	49	168	63	42
Point B average	87	96	49	80	94	40
Point C average	64	87	116	54	82	114
Point D average	137	53	90	137	45	85
Point E average	71	92	64	63	88	58

Table 15 – Substituted values in figure 14

III. Appendix C

Script used in chapters 4.1 and 4.2:

```
close all
clc
clear

lin = 768;
col = 1024;

%% load data into 'y(row,colomn,time_index)', 'u(row,colomn,time_index)'
and 'v (row,colomn,time_index)' variables

load(y,u,v)


%% divide the data into a matrix 6*8
pool=199-9;
time=5:16/190:21-16/190;
sumy=zeros(6,8,pool);
sumu=zeros(6,8,pool);
sumv=zeros(6,8,pool);
for k=1:pool
    for x=0:5
        for z=0:7
            for i=1+x*128:128+128*x
                for j=1+z*128:128+128*z
                    sumy(1+x,1+z,k)=sumy(1+x,1+z,k) + y(i,j,k+9);
                    sumu(1+x,1+z,k)=sumu(1+x,1+z,k) + u(i,j,k+9);
                    sumv(1+x,1+z,k)=sumv(1+x,1+z,k) + v(i,j,k+9);
                end
            end
        end
    end
end



sumy = floor(sumy./128./128);
sumu = floor(sumu./128./128);
sumv = floor(sumv./128./128);
```

```
%prints data

wind=zeros(1,pool);
figure
for z=0:7
    for x=0:5
        for k=1:pool
            subplot(6,8,z+x*8+1)
            wind(k)=sumy(1+x,1+z,k);
            plot(time, wind)
            axis([5 21 0 200])
        end
    end
end



%% calculate average values for each of the timelapses
 graphy=zeros(1,pool);
graphu=zeros(1,pool);
graphv=zeros(1,pool);
for k=1:pool
    graphy(k)=sum(sum(y(:,:,k)));
    graphu(k)=sum(sum(u(:,:,k)));
    graphv(k)=sum(sum(v(:,:,k)));
end

graphy=graphy/lin/col;
graphu=graphu/lin/col;
graphv=graphv/lin/col;

figure
plot(time,graphy)
grid on
axis([5 21 0 140])
ylabel('Y level')
xlabel('Time (hours)')
figure
plot(time,graphu)
grid on
axis([5 21 123 133])
ylabel('U level')
xlabel('Time (hours)')
figure
plot(time,graphv)
grid on
axis([5 21 123 133])
ylabel('V level')
xlabel('Time (hours)')
```

IV. Appendix D

Script used in chapter 3.2

```
close all
clc
clear

lin = 768;
col = 1024;

%%

y_cores_max=importdata('y_cores_max.txt');
u_cores_max=importdata('u_cores_max.txt');
v_cores_max=importdata('v_cores_max.txt');

imshow(uint8(y_cores_max(150:610,1:865)))

amarelo=y_cores_max(170:360,30:240);
imshow(uint8(amarelo))
amarelo=sum(sum(amarelo))/191/211

uamarelo=u_cores_max(170:360,30:240);
uamarelo=sum(sum(uamarelo))/191/211

vamarelo=v_cores_max(170:360,30:240);
vamarelo=sum(sum(vamarelo))/191/211



laranja=y_cores_max(180:382,265:460);
imshow(uint8(laranja))
laranja=sum(sum(laranja))/203/196

ularanja=u_cores_max(180:382,265:460);
ularanja=sum(sum(ularanja))/203/196

vlaranja=v_cores_max(180:382,265:460);
vlaranja=sum(sum(vlaranja))/203/196
```

```
vermelho=y_cores_max(190:390,470:650);
imshow(uint8(vermelho))
vermelho=sum(sum(vermelho))/201/181

uvermelho=u_cores_max(190:390,470:650);
uvermelho=sum(sum(uvermelho))/201/181

vvermelho=v_cores_max(190:390,470:650);
vvermelho=sum(sum(vvermelho))/201/181



verde=y_cores_max(205:400,680:850);
imshow(uint8(verde))
verde=sum(sum(verde))/196/171

uverde=u_cores_max(205:400,680:850);
uverde=sum(sum(uverde))/196/171

vverde=v_cores_max(205:400,680:850);
vverde=sum(sum(vverde))/196/171



azul=y_cores_max(390:580,40:230);
imshow(uint8(azul))
azul=sum(sum(azul))/191/191

uazul=u_cores_max(390:580,40:230);
uazul=sum(sum(uazul))/191/191

vazul=v_cores_max(390:580,40:230);
vazul=sum(sum(vazul))/191/191



violeta=y_cores_max(400:580,260:450);
imshow(uint8(violeta))
violeta=sum(sum(violeta))/181/191

uvioleta=u_cores_max(400:580,260:450);
uvioleta=sum(sum(uvioleta))/181/191

vvioleta=v_cores_max(400:580,260:450);
vvioleta=sum(sum(vvioleta))/181/191



preto=y_cores_max(410:580,490:650);
imshow(uint8(preto))
preto=sum(sum(preto))/171/161

upreto=u_cores_max(410:580,490:650);
upreto=sum(sum(upreto))/171/161
```

```
vpreto=v_cores_max(410:580,490:650);
vpreto=sum(sum(vpreto))/171/161



branco=y_cores_max(420:580,672:780);
imshow(uint8(branco))
branco=sum(sum(branco))/161/109

ubranco=u_cores_max(420:580,672:780);
ubranco=sum(sum(ubranco))/161/109

vbranco=v_cores_max(420:580,672:780);
vbranco=sum(sum(vbranco))/161/109
```

V. Appendix E

Script used in chapter 3.1

```
clc
clear
close all

y=importdata('yCam.txt');
u=importdata('uCam.txt');
v=importdata('vCam.txt');

y=y(15:248,20:250);
u=u(15:248,20:250);
v=v(15:248,20:250);

r=importdata('rCam.txt');
g=importdata('gCam.txt');
b=importdata('bCam.txt');

ry=importdata('ryCam.txt');
gy=importdata('gyCam.txt');
by=importdata('byCam.txt');


% r(50:52,40:42)=0;
% g(50:52,40:42)=0;
% b(50:52,40:42)=0;
%
% r(150:152,50:52)=0;
% g(150:152,50:52)=0;
% b(150:152,50:52)=0;
%
% r(170:172,210:212)=0;
% g(170:172,210:212)=0;
% b(170:172,210:212)=0;
%
% r(40:42,190:192)=0;
% g(40:42,190:192)=0;
% b(40:42,190:192)=0;
%
% r(180:182,120:122)=0;
% g(180:182,120:122)=0;
% b(180:182,120:122)=0;
```

```
mean2(r(180:182,120:122))
mean2(g(180:182,120:122))
mean2(b(180:182,120:122))
mean2(ry(180:182,120:122))
mean2(gy(180:182,120:122))
mean2(by(180:182,120:122))


r(50:52,40:42)=ry(50:52,40:42);
g(50:52,40:42)=gy(50:52,40:42);
b(50:52,40:42)=by(50:52,40:42);

r(150:152,50:52)=ry(150:152,50:52);
g(150:152,50:52)=gy(150:152,50:52);
b(150:152,50:52)=by(150:152,50:52);

r(170:172,210:212)=ry(170:172,210:212);
g(170:172,210:212)=gy(170:172,210:212);
b(170:172,210:212)=by(170:172,210:212);

r(40:42,190:192)=ry(40:42,190:192);
g(40:42,190:192)=gy(40:42,190:192);
b(40:42,190:192)=by(40:42,190:192);

r(180:182,120:122)=ry(180:182,120:122);
g(180:182,120:122)=gy(180:182,120:122);
b(180:182,120:122)=by(180:182,120:122);


A=y(51,41)/256
A=u(51,41)/256
A=v(51,41)/256

B=y(151,51)/256
B=u(151,51)/256
B=v(151,51)/256

C=y(171,211)/256
C=u(171,211)/256
C=v(171,211)/256

D=y(41,191)/256
D=u(41,191)/256
D=v(41,191)/256

E=y(181,121)/256
E=u(181,121)/256
E=v(181,121)/256



X(:,:,1)=r(15:248,20:250);
X(:,:,2)=g(15:248,20:250);
X(:,:,3)=b(15:248,20:250);
X=uint8(X);
```

```
figure
imshow(X)


XX(:,:,1)=ry(15:248,20:250);
XX(:,:,2)=gy(15:248,20:250);
XX(:,:,3)=by(15:248,20:250);
XX=uint8(XX);
figure
imshow(XX)


%%

q=ry-r;
w=gy-g;
e=by-b;

figure
histogram(q)
xlabel('Deviation')
ylabel('Number of Occurences')
figure
histogram(w)
xlabel('Deviation')
ylabel('Number of Occurences')
figure
histogram(e)
xlabel('Deviation')
ylabel('Number of Occurences')


%%

clear
clc



y=importdata('yCam.txt');
u=importdata('uCam.txt');
v=importdata('vCam.txt');

y=y(15:248,20:250);
u=u(15:248,20:250);
v=v(15:248,20:250);


r=importdata('rCam.txt');
g=importdata('gCam.txt');
b=importdata('bCam.txt');


bri=0.299*r+0.587*g+0.114*b;
```

```
bri=bri(15:248,20:250);


menos = y-bri;

m=round(menos);

[a b c] = unique(m);
zz=[m(b) histc(c, 1:max(c))]

B = [m(b) histc(c, 1:max(c))];
stem(B(:,2), 'Linewidth', 2, 'Marker', 'none')
set(gca, 'XTickLabel', B(:,1))
axis([0 size(B,1)+1 0 1.1*max(B(:,2))])
box off
```

VI. Appendix F

Script used in chapter 3.3

```
close all
clc
clear

lin = 768;
col = 1024;

%%

%200
y(:,:,1)=importdata('y_13_36_27.txt');
u(:,:,1)=importdata('u_13_36_27.txt');
v(:,:,1)=importdata('v_13_36_27.txt');

%180
y(:,:,2)=importdata('y_13_41_53.txt');
u(:,:,2)=importdata('u_13_41_53.txt');
v(:,:,2)=importdata('v_13_41_53.txt');

%160
y(:,:,3)=importdata('y_13_45_18.txt');
u(:,:,3)=importdata('u_13_45_18.txt');
v(:,:,3)=importdata('v_13_45_18.txt');

%140
y(:,:,4)=importdata('y_13_48_18.txt');
u(:,:,4)=importdata('u_13_48_18.txt');
v(:,:,4)=importdata('v_13_48_18.txt');

%120
y(:,:,5)=importdata('y_13_52_12.txt');
u(:,:,5)=importdata('u_13_52_12.txt');
v(:,:,5)=importdata('v_13_52_12.txt');

%100
y(:,:,6)=importdata('y_13_57_55.txt');
u(:,:,6)=importdata('u_13_57_55.txt');
v(:,:,6)=importdata('v_13_57_55.txt');
```

```
%80
y(:,:,7)=importdata('y_14_5_23.txt');
u(:,:,7)=importdata('u_14_5_23.txt');
v(:,:,7)=importdata('v_14_5_23.txt');

%60
y(:,:,8)=importdata('y_14_13_14.txt');
u(:,:,8)=importdata('u_14_13_14.txt');
v(:,:,8)=importdata('v_14_13_14.txt');

%50
y(:,:,9)=importdata('y_14_16_37.txt');
u(:,:,9)=importdata('u_14_16_37.txt');
v(:,:,9)=importdata('v_14_16_37.txt');

%40
y(:,:,10)=importdata('y_14_22_46.txt');
u(:,:,10)=importdata('u_14_22_46.txt');
v(:,:,10)=importdata('v_14_22_46.txt');

%30
y(:,:,11)=importdata('y_14_37_43.txt');
u(:,:,11)=importdata('u_14_37_43.txt');
v(:,:,11)=importdata('v_14_37_43.txt');

%25
y(:,:,12)=importdata('y_16_1_36.txt');
u(:,:,12)=importdata('u_16_1_36.txt');
v(:,:,12)=importdata('v_16_1_36.txt');



%%


pool=12;
sumy=zeros(pool,1);
sumu=zeros(pool,1);
sumv=zeros(pool,1);

sumy2=zeros(pool,1);
sumu2=zeros(pool,1);
sumv2=zeros(pool,1);
notimes=0;

for k=1:pool
    a=y(430:625,540:770,k);
    sumy(k)=sum(sum(a));
    a=u(430:625,540:770,k);
    sumu(k)=sum(sum(a));
    a=v(430:625,540:770,k);
    sumv(k)=sum(sum(a));
    for i=1:lin
        for j=1:col
            if (i<430 || i>625) || (j<540 || j>770)
```

```
                sumy2(k) = sumy2(k) + y(i,j,k);
                sumu2(k) = sumu2(k) + u(i,j,k);
                sumv2(k) = sumv2(k) + v(i,j,k);
                if k==1
                notimes=notimes+1;
                end
            end
        end
    end
end

%% heat rock
sumy = floor(sumy./196./234);
sumu = floor(sumu./196./234);
sumv = floor(sumv./196./234);

figure
plot(sumy)
figure
plot(sumu)
figure
plot(sumv)

%% rest of scene
sumy2 = floor(sumy2./notimes);
sumu2 = floor(sumu2./notimes);
sumv2 = floor(sumv2./notimes);

figure
plot(sumy2)
figure
plot(sumu2)
figure
plot(sumv2)
```